Titulo: *e-CRF Manager*

Volumen:

Alumno: LUISA MIMUN HERNANDO

Director/Ponente: JOSÉ ANTONIO GONZÁLEZ ALASTRUE

Departamento: Estadística.

Fecha: viernes 3 de febrero 2012

DATOS DEL PROYECTO

TÍTULO: e-CRF Manager

AUTOR: LUISA MIMUN HERNANDO luisa.mimun@est.fib.upc.edu
FECHA DE INSCRIPCIÓN: VIERNES 3 DE FEBRERO 2012

DIRECTOR: JOSÉ ANTONIO GONZÁLEZ ALASTRUE jose.a.gonzalez@upc.edu
DEPARTAMENTO DEL DIRECTOR: ESTADÍSTICA

TITULACIÓN: INGENIERIA TÈCNICA EN INFORMATICA DE SISTEMAS.
CRÉDITOS : 22.5
CENTRO: *Facultat d'Informàtica de Barcelona (FIB)*
Universitat: Universitat Politècnica de Catalunya (UPC)
BarcelonaTech

CO-DIRECTOR: JUAN VICENTE TORRES MARTIN jvtorres@dm-stat.com
DEPARTAMENTO DEL DIRECTOR:

TRIBUNAL

DIREDTOR/A: JOSÉ ANTONIO GONZÁLEZ ALASTRUE jose.a.gonzalez@upc.edu
PRESIDENTE/A: ERIK COBO VALERI erik.cobo@upc.edu
VOCAL: ALBERT OLIVERAS LLUNELL oliveras@lsi.upc.edu

ÍNDICE

CONTENIDO

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

1.1. DESCRIPCIÓN GENERAL DEL PROYECTO

Un ensayo clínico es una investigación que se realiza en seres humanos con el propósito de evaluar la eficacia de una intervención médica o quirúrgica.

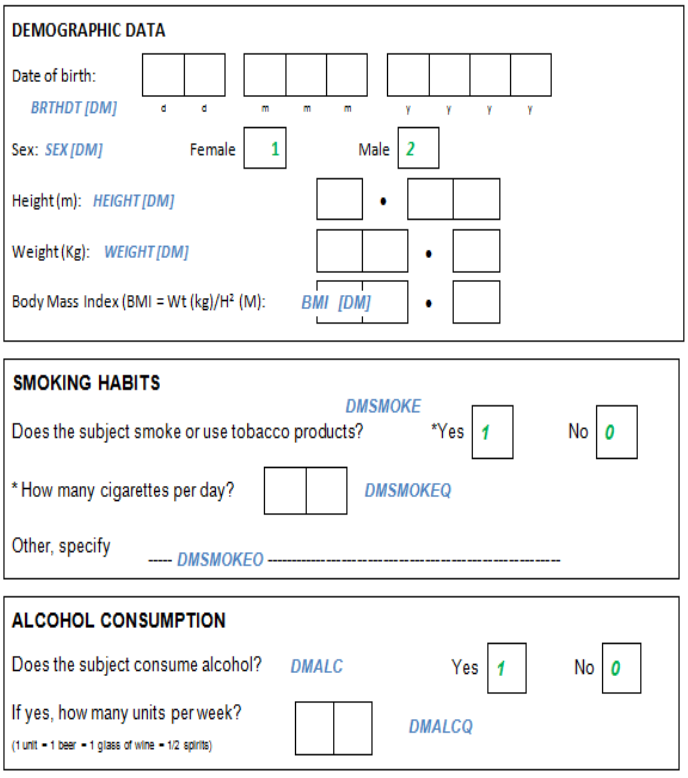
Su desarrollo se realiza mediante una plantilla de hoja de cálculo EXCEL con el propósito de crear un registro electrónico Cuaderno de Recogida de Datos (CRF), documento con el que el investigador registra una serie de datos necesarios (demográficos, historia clínica, medicación y eventos adversos.) de cada paciente o grupo de pacientes durante la realización del estudio o ensayo clínico en un centro. El ensayo incorpora evaluaciones médicas específicas que pueden repetirse a lo largo del tiempo, mediante la programación de diferentes visitas, para evaluar con los datos obtenidos si una intervención produce un cambio en la salud.

La plantilla permite crear un libro a partir de las características propias para cada ensayo.
El libro EXCEL creado permite disponer del Registro de Inclusión del Estudio.

ANEXO- 1 INTRODUCTION

A modo de ejemplo, la siguiente figura representa una página común del CRF. Esta página está compuesta por tres secciones:

- Datos demográficos.
- Hábitos de fumar.
- El consumo de alcohol.



2. OBJETIVOS DEL PROYECTO

- Desarrollar una plantilla de hoja de cálculo EXCEL con el propósito de crear un registro electrónico Cuaderno de Recogida de Datos (CRF) para un ensayo clínico. (Obj1)[1]

- Poder crear y gestionar una aplicación en un entorno Web sin la necesidad de generar una gran carga de trabajo de programación y qué permita:(Obj2)[2]

    - Ofrecer el registro y control de usuarios administrador e investigador. Según el rol del usuario, realizar diferentes funcionalidades dentro de la aplicación.(Obj2.1)

    - Crear un estudio a partir de la lectura de la plantilla de EXCEL.Y visualizarla de forma ordenada.(Obj2.2)

    - Poder Gestionar la creación, modificación y eliminación de pacientes del ensayo clínico.(Obj 2.3)

- Se quiere conseguir que la aplicación sea independiente del sistema que utilice y sea lo más portable posible.(Obj3)

- Conocer la normativa y procedimientos vigentes, en todo lo relacionado con las medidas de seguridad, implementadas en los sistemas de tratamientos de datos de pacientes. (Obj4)

- Otro objetivo para el desarrollo e implementación de la aplicación es la utilización de Software Libre.(Obj5)

[1] Objetivo [número].

3. MOTIVACIÓN

Los motivos por los que escogí este proyecto, son poder aplicar los conocimientos adquiridos durante la carrera y ayudar en lo posible en el desarrollo de aplicaciones para estudios médicos de investigación con personas, que ayuden a la sociedad a avanzar en la recogida de datos médicos que sirvan de indicadores para examinar nuevas formas de prevenir, detectar, diagnosticar, tratar enfermedades y mejorar aspectos en la salud de los pacientes. Poder participar en una aplicación destinada a facilitar la captación de datos durante los ensayos clínicos, ofreciendo seguridad y trazabilidad de los procedimientos realizados así como incrementar la confidencialidad y seguridad de los datos almacenados.

Las necesidades del proyecto hacen posible poner en práctica muchos conceptos adquiridos a lo largo de las asignaturas impartidas durante la carrera ya que hará uso de entornos y lenguajes que se utilizan en la actualidad y utilizar un entorno web abre la posibilidad de tener ofertas en el mercado laboral, ya que son tecnologías utilizadas en muchos sectores especialmente en los de servicios y nuevas tecnologías.

El tener la oportunidad de desarrollar este proyecto real facilitará el conocimiento para desarrollarme profesionalmente.

4. ANÁLISIS DE REQUERIMIENTOS

4.1. ESPECIFICACIÓN DE REQUERIMIENTOS

El sistema se basa en un conjunto de requerimientos funcionales, que hacen posible que el sistema funcione de forma correcta.
Los requerimientos son los siguientes:

R01	Definición de las hojas del EXCEL Define.xls
El Define.xls se organizará en cuatro hojas de cálculo: (1) Información General. *General Info (GI)* (2) Las secciones de distribución. *Sections distribution (SDIS)* (3) Definición de las secciones. *Sections definition (SDEF)* (4) Los formatos**.** *Formats (FM*)	
Prerrequisitos: ninguno	Valoración:

Ilustración 2-Requerimiento -R01-Definición de las hojas del EXCEL Define.xls

R02	Composición de la hoja *General Info (GI)* del EXCEL Define.xls
En esta hoja de cálculo se deberán definir los datos del cliente, el código del cliente, el protocolo, el nombre y además incluir un nombre corto para el estudio.	
Prerrequisitos: R01	Valoración:

Ilustración 3- Requerimiento -R02-Composición de la hoja General Info (GI) del EXCEL Define.xls

R03	Composición de la hoja *Sections distribution (SDIS)* del EXCEL Define.xls
En esta hoja de cálculo se definirá cómo serán distribuidas las secciones del Cuaderno de Recogida de Datos (CRF) en diferentes páginas. Las secciones deberán identificarse con un título y un nombre abreviado. Las secciones podrán repetirse a lo largo del estudio con diferentes números de página. Cada sección deberán obligatoriamente estar relacionada con un nombre de visita (numérico o texto). En cada sección se deberá indicar obligatoriamente en qué página se presentará.	
Prerrequisitos: R01	Valoración:

Ilustración 4-Requerimiento-R03-Composición de la hoja Sections distribution (SDIS)del EXCEL Define.xls

R04	Composición de la hoja *Sections definition (SDEF)* del EXCEL Define.xls
En esta hoja de cálculo se define el contenido de las secciones. Cada sección deberá estar definida sólo una vez. Una sección deberá estar formada por estructuras y variables. La Sección contendrá obligatoriamente el nombre de la misma sección para las variables que formarán la sección. Se deberán definir obligatoriamente cuatro estructuras diferentes: línea, panel, sección y tabla. Los nombres de las variables deberán tener las dos primeras letras de la tabla a la que pertenecen y no exceder de 8 caracteres de longitud. Este campo deberá ser obligatorio. Para completar la información de las variables se deberán obligatoriamente definir las columnas con los nombres de las Tablas de la Base de Datos, la columna con los textos de la pregunta donde se mostrarán en el formulario web al lado izquierdo del campo, este campo podrá ser opcional y otra columna con el texto para mostrar en el lado derecho del campo, este campo también puede ser opcional. Y deberá contener dos columnas mas una con el tipo de tipo de campo y otra con el formato.	
Prerrequisitos: R01-R03	Valoración:

Ilustración 5- Requerimiento-R04- Composición de la hoja Sections definition (SDEF) del EXCEL Define.xls

R05	Composición de la hoja *Formats (FM)* del EXCEL Define.xls
Se deberán incluir en esta hoja de cálculo, los formatos correspondientes, los valores y etiquetas.	
Prerrequisitos: R01-R04	Valoración:

Ilustración 6- Requerimiento-R05-Composición de la hoja Formats (FM) del EXCEL Define.xls

R06	Control de acceso al sistema
Los usuarios que entren al portal Web deberán autenticarse. Los datos de autenticación serán: login y password.	
Prerrequisitos: ninguno	Valoración:

Ilustración 7- Requerimiento-R03-Control de acceso al sistema

Al aspecto visual del sistema no se le ha dedicado mucho tiempo, ya que no es uno de nuestros objetivos principales, pero si se ha dedicado mucho más tiempo y esfuerzo a la parte de definición y de funcionalidad de la aplicación para hacerla lo más robusta y eficaz posible.

4.1.1. ANÁLISIS FUNCIONAL DEL SISTEMA

El sistema debe proporcionar una serie de funcionalidades dependiendo del usuario que se conecte o participe y del rol definido. Hemos distinguido dos actores: Administrador e Investigador.
El sistema web estará compuesto por dos secciones bien diferenciadas, el portal de control donde tendrá acceso el administrador y el portal accesible para el investigador.

4.1.1.1. FUNCIONALIDADES DEL ADMINISTRADOR

El administrador también hará uso del Software y opciones de gestión instaladas en el Servidor Web al cual sólo tiene acceso el usuario administrador.

Las funciones básicas que deberá realizar el administrador son:

RFA01	Manejar Usuarios
El administrador deberá gestionar usuarios del sistema. Crear, ver, editar y eliminar usuarios.	
Prerrequisitos: ninguno	Valoración:

Ilustración 8- Requerimiento-RFA01-Manejar Usuarios

RFA02	Manejar Estudios
El administrador podrá crear un estudio a partir de la lectura de la plantilla de EXCEL Define.xls. Y visualizarlo de forma ordenada. El administrador podrá exportar el estudio a una plantilla de EXCEL Define.xls.	
Prerrequisitos: R01	Valoración:

Ilustración 9- Requerimiento-RFA02-Manejar Estudios

4.1.1.2. FUNCIONALIDADES DEL INVESTIGADOR

Las funciones básicas del investigador serán:

RFI01	Manejar Pacientes
El Investigador deberá gestionar pacientes que participen en el ensayo clínico. Crear, ver, editar y eliminar pacientes. De los pacientes se recogerán únicamente las iniciales.	
Prerrequisitos: RFA01	Valoración:

Ilustración 10- Requerimiento-RFI01-Manejar Pacientes

RFI02	Manejar el Cuaderno de Recogida de Datos(CRF)
El Investigador deberá ver el Cuaderno de Recogida de Datos (CRF) .	
Prerrequisitos: RFA02	Valoración:

Ilustración 11- Requerimiento-RFI02-Manejar el Cuaderno de Recogida de Datos (CRF)

4.2. REQUERIMIENTOS NO FUNCIONALES

Los requerimientos no funcionales definen las cualidades o propiedades que el sistema debe cumplir como pueden ser la seguridad, la escalabilidad, la eficiencia y la portabilidad.

Referente a los aspectos legales de utilización de software hemos verificado que las herramientas que vamos a utilizar dispongan de licencia vigente. Con esto nos aseguramos cumplir con todos los aspectos legales de utilización del software, tecnologías y librerías usadas para la elaboración de la aplicación.

El sistema no recogerá datos médicos de pacientes, por este motivo no hemos profundizado en ley de protección de datos. Pero si que en versiones posteriores se tendrán que tener en cuenta las normas u políticas nacionales e internacionales para asegurar que las investigaciones que se realizan con personas se llevan a cabo de acuerdo a principios científicos y éticos.

4.2.1. SEGURIDAD

El sistema es robusto desde el principio, de este modo nos aseguramos que las personas no autorizadas tengan acceso, la identificación y reconocimiento de la sesión son obligatorias para acceder al sistema.
Los administradores tienen acceso a gestionar usuarios y sus privilegios para asegurarnos de la confidencialidad de la clave, solo los usuarios conocen su contraseña ya que esta codificada mediante el algoritmo MD5[3], de esta forma nos aseguramos que ninguna persona pueda ejecutar funcionalidades para las cuales no tiene privilegios.

También se analizan las debilidades de las herramientas que hemos utilizado y los tipos de seguridad que ofrece para la protección como en el caso del Servidor Web.
Para chequear la seguridad en la herramienta Xampp podemos a través de la página de inicio, administrar el Chequeo de Seguridad donde podemos apreciar las vulnerabilidades que presenta la herramienta. Podemos estudiar los puntos inseguros y corregirlos.

[3] En criptografía, **MD5** (abreviatura de *Message-Digest Algorithm 5*, Algoritmo de Resumen del Mensaje es un algoritmo de reducción criptográfico de 128 bits ampliamente usado.

El manejo de Claves en Xampp permite administrar claves a través de un panel de administración, donde se puede gestionar la contraseña del usuario *root*[4], para mantener la integridad de la información que se almacena en la Base de Datos. También permite cambiar la contraseña en el fichero *config.ini.php* situado en */phpMyAdmin/.*

4.2.2. ESCALABILIDAD

El proyecto tiene posibles mejoras en aspecto visual, pero se realiza de forma que pueda ser modificado de forma fácil. La aplicación sé ha basado en la utilización del patrón MVC[5] para separar el interfaz de usuarios, permitiendo su fácil modificación y restructuración. Esto facilitará el análisis de nuevas funcionalidades permitiendo la ampliación y mejora de funcionalidades en futuras versiones.

4.2.3. EFICIENCIA

El sistema garantiza que los usuarios puedan utilizar la plataforma con un rendimiento óptimo sin tiempos de respuesta excesivos. Se garantiza que el administrador y el investigador puedan trabajar con un sistema ágil y fiable. No hemos encontrado muchos inconvenientes de eficiencia a la hora de implementar el código.

4.2.4. PORTABILIDAD

Un punto fuerte del sistema es su versatilidad por su portabilidad y ampliación con otras plataformas. La Web se realizo en el entorno de pruebas con un Sistema Operativo Windows y no supuso ningún problema el cambio a la plataforma ofrecida por la Universidad que ha sido Linux. El único requisito era que el Sistema Operativo tuviera instalado un Servidor Web y un Servidor de Base de Datos.

[4] En sistemas operativos, **root** es el nombre convencional de la cuenta de usuario que posee todos los derechos en todos los modos.

[5] **Modelo Vista Controlador (MVC)** es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de negocio en tres componentes distintos.

5. SOLUCIÓN PROPUESTA

5.1. PLANIFICACIÓN DEL PROYECTO

La carga de trabajo se ha estimado en 20 horas por crédito. De esta manera el proyecto que consta de 22,5 créditos se está llevando a cabo durante un cuatrimestre a tiempo parcial (20 horas a la semana) un total de 450 horas.
Para realizar el proyecto, se elaboró una planificación global lo más aproximada a la realidad para administrar el tiempo y el trabajo por hacer, se realizó un Diagrama de Gantt.
Las tareas se obtuvieron de las especificaciones y se fijaron cronológicamente en la línea de tiempo asignando y ajustando la fecha de inicio y la de fin. Algunas se han dividido en otras rutinas para hacer un grado más de detalle y poder dividir mejor el trabajo. Además se han realizado asociaciones entre tareas de manera que no se empezaba una tarea hasta que no se acaba la anterior.
El grupo se ha organizado de forma que el director supervisaba las tareas, para dirigirlas de forma ordenada, de revisar las finalizaciones de las tareas y reclamar aquellas actividades que pudieran llegar a ser un posible camino crítico. Si la tarea era muy extensa se realizaban ajustes necesarios para llevar a cabo la tarea a tiempo.
Las posibles rutas críticas serían aquellas actividades que debían realizarse a tiempo, como podía ser el objetivo 2 con la decisión de la utilización y elección del Framework ya que tuvimos que leer manuales y tutoriales.

Se están cubriendo todas las necesidades para poder controlar y gestionar el tiempo del que disponemos. Durante el tiempo que queda se pondrá a prueba la aplicación para mejorar y finalizar el objetivo 2 con las mejoras de visualización del Cuaderno de Recogida de Datos.

Las fases del proyecto se han dividido en, una fase inicial donde nos hemos centrado en analizar y reunir todos los requerimientos de la aplicación. En esta fase se ha analizado el servicio que queríamos ofrecer y los diferentes perfiles que usarían la aplicación.
En la fase de elaboración una de las primeras decisiones fue la elección de las tecnologías que utilizaríamos para desarrollar el proyecto y analizar los diferentes Frameworks.
En la fase de Construcción se implementó la arquitectura con las funcionalidades definidas en la fase anterior. Y en la fase de transición se migrará al Servidor de producción ofrecido por la Universidad.

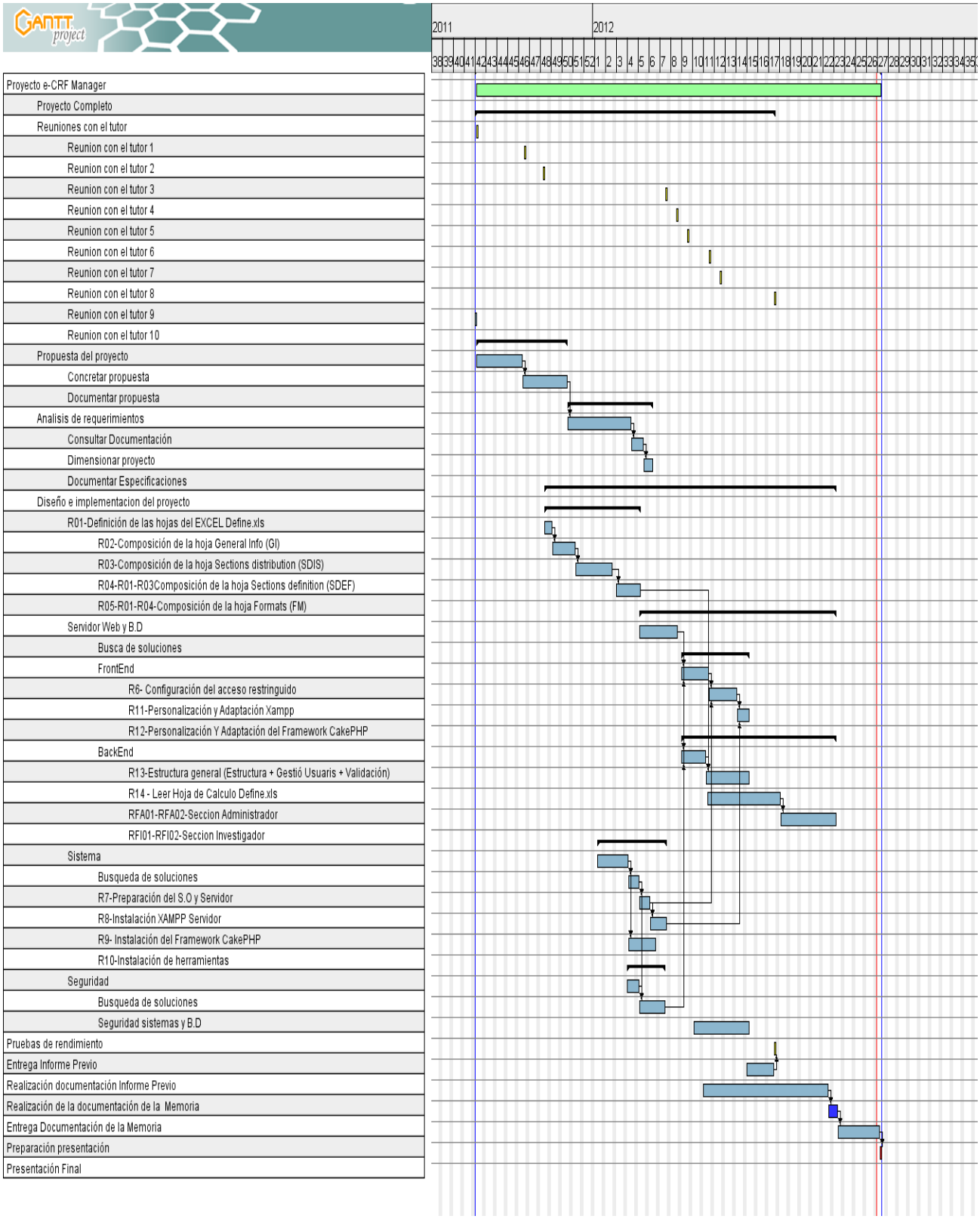


Ilustración 12- Planificación del Proyecto -Gantt Project

Durante los días previos a la exposición se pondrá a prueba la aplicación para mejorar su funcionamiento y se procederá a acabar la documentación necesaria para la entrega de la memoria y los anexos con sus respectivas mejoras y/o cambios.

5.2. HERRAMIENTAS UTILIZADAS

5.2.1. ARQUITECTURA DEL SISTEMA Y PLATAFORMA

Los Sistemas Operativos que utilizaremos en el proyecto en el entorno de preproducción vamos a utilizar Windows y para el entorno de producción o alojamiento vamos a utilizar Linux.

5.2.2. CONSIDERACIONES INICIALES DE ELECCIÓN.

El punto de partida para cualquier creación de una aplicación Web es analizar cuál será el entorno donde se ubicará la Web, es decir, las prestaciones que ofrecerán el Servidor y las características que queremos para la propia aplicación. El Servidor donde se alojará el portal, será el de la Universidad. Esta Facultad con sede en Barcelona, nos ofrece un Servidor Web y un Servidor de Base de Datos.

Las consideraciones iniciales trataban de elegir con que lenguaje de programación realizaríamos la aplicación Web. Los lenguajes de programación más extendidos son Active Server Pages (ASP) y Hypertext Preprocessor (PHP). Con los dos lenguajes se pueden realizar las mismas aplicaciones, las propiedades y modo de funcionamiento son muy similares. Los dos lenguajes se ejecutan mediante scripts que serán interpretados por el Servidor, facilitan el acceso y comunicación con Bases de Datos. El hecho diferencial de PHP respecto a ASP es que al ser Software Libre, conlleva la aparición de módulos o plug-ins específicos realizados por la comunidad que aportan cualquier función o característica de forma gratuita. Por el contrario ASP se apoya sobre librerías o módulos que deben ser adquiridos a las empresas privadas que los crean. Él lenguaje ASP es más sencillo de interpretar, pero a la hora de programar e implementar resulta más complicado, sobre todo comparándolo con PHP.

Otra consideración para la realización de la aplicación Web era ver las posibilidades existentes en el mundo de los gestores de bases de datos, sistemas pensados para servir de interfaz entre la Base de Datos, el usuario y las aplicaciones. Una de las características que tendremos en cuenta en la elección, es que se trate de un software gratuito. Los gestores Open Source más tradicionales son PostgreSQL, MySQL y SQLite.

Otra consideración importante era decidirse si utilizaríamos un Framework que nos ayudara en el desarrollo de la Web, sin necesidad de realizar una gran programación, ya que crear una solución partiendo de cero, puede ser muy costoso.
Los Frameworks nacen de la necesidad de crear proyectos organizados y aportar herramientas de desarrollo que agilizan este proceso. Un Framework ya aporta multitud de módulos de software creados previamente, fácilmente acoplables a nuestro proyecto en caso de necesidad. Todas estas aportaciones permiten a los diseñadores y programadores centrar el mayor tiempo posible en identificar requerimientos en vez de discutir problemas de bajo nivel. Como inconveniente, nos

podemos encontrar que hay Frameworks que introducen excesivas líneas de código innecesarias que hacen el código más difícil de leer para un programador no familiarizado con el entorno.
De los Frameworks existentes encontramos el Frameworks Java con Hibernate, Frameworks PHP con CakePHP, Symphony y CodeIgniter; Y Frameworks Python con Django.
De entre ellos valoramos que Framework se ajusta mejor a nuestras necesidades y escogimos la utilización del Frameworks PHP, CakePHP o Synphony.
Para decidirnos por uno de los dos escogidos se propuso realizar una votación sencilla, seleccionando una serie de cualidades que nos ayudaran a decidirnos por uno y siempre que fuera posible se votaba al mejor de los dos, es decir asignamos un + al mejor y un - al peor. El que tuviera más puntos positivos ganaba.

Valor(1-2)	Cualidad	Cakephp Final:2.0.6	vs	Synfony 1.4
		Nota (+ al mejor y un - al peor)		Nota (+ al mejor y un - al peor)
2	Rapidez de ejecución	+		-
1	Fácil Instalación y configurar	+		-
2	Reglas de validación en cliente JS	+		-
2	Reglas de validación en servidor PHP	+		-
1	Disponibilidad en hostings (bluehost y altervista)	-		+
1	Simple de usar	+		-
1	Curva de aprendizaje	-		+
2	Orientada a objetos	+		-
2	Estable	+		-
2	Código legible	+		-
2	Fácil de extender	-		+
2	Pruebas y depuraciones	+		-
2	Uso de plantillas	+		+
2	Comunidad activa de usuarios	-		+
2	Asistente de Vistas	-		+
2	Manejo de memoria caché	+		-
2	Mecanismos de autentificación y credenciales	+		-
2	URLs inteligentes	-		+
2	Compatibilidad con php	+		-
2	Construcción de vistas	+		-
2	Creación de formularios	+		-
2	Excels	-		+
	SUMA	28		14

Ilustración 13 - Votación Cualidades del Framework

5.2.3. DESCRIPCIÓN DE LOS LENGUAJES DE PROGRAMACIÓN UTILIZADOS

5.2.3.1. DESCRIPCIÓN DEL SERVIDOR WEB

APACHE 

El servidor web escogido, para la instalación en el ordenador local, de la Web será APACHE, ya que es el servidor HTTP más utilizado en la red, y además es un software libre.

Las características del servidor web son las siguientes:

- Servidor Web muy fiable.
- De fácil instalación.
- Actualización constante de protocolos, por tratarse de software libre.
- Posibilidad de añadir módulos para incrementar su compatibilidad y funciones.
- Funciona en todos los Sistemas Operativos.
- Gran integración con lenguajes PHP y MySQL.

Una vez terminado el trabajo en el ordenador local, se trasladará al servidor de la Universidad, que dispone de Servidor Web propio con una serie de características.

5.2.3.2. DESCRIPCIÓN DE LOS LENGUAJES

HTML 

HTML [6]es un lenguaje de marcado que deriva del SGML diseñado para estructurar textos y relacionarlos en forma de hipertexto. Desarrollado por el consorcio W3C[7].

Las ventajas del lenguaje son las siguientes:

- Sencillo, permite describir hipertexto.
- Texto presentado de forma estructurada y agradable.
- Ficheros de poco tamaño.
- Ejecución rápida. Lenguaje fácil de aprender.
- El admiten todos los exploradores.

[6] (Acrónimo de Hyper Text Markup Language, en castellano, "lenguaje de marcado de hipertexto"),.

[7] (World Wide Web Consortium)

Las desventajas del lenguaje son las siguientes:

- Lenguaje estático.
- La interpretación de los navegadores puede ser diferente.
- Guarda muchas etiquetas que pueden convertirse en inútiles y dificultan la corrección.
- El diseño es más lento.
- Las etiquetas son muy limitadas.

PHP 

El lenguaje escogido para programar la Web será PHP principalmente por sus ventajas.

Las ventajas del lenguaje son las siguientes:

- Es un lenguaje muy sencillo de aprender.
- Se caracteriza por ser un lenguaje muy rápido.
- Es el lenguaje de programación más utilizado para generar páginas web de forma dinámica.
- Soporta en cierta medida la orientación a objetos, clases y herencias.
- Es un lenguaje multiplataforma compatible con los principales Sistemas Operativos Linux, Windows, MAC entre otros y con los principales servidores Web (Apache, Microsoft IIS, Netscape, etc.).Capacidad de conexión con la mayoría de los gestores de bases de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, entre otros.
- Capacidad de expandir su potencial utilizando módulos.
- Herramienta de código libre o Open Source, por lo que representa con una alternativa de fácil acceso para todos.
- No requiere definición de tipos de variables ni gestión detallada de bajo nivel.

Las desventajas del lenguaje son las siguientes:

- Necesita de la instalación de un Servidor Web.
- La programación orientada a objetos se vuelve ineficiente en aplicaciones grandes.
- Dificulta la modularización.
- Dificulta la organización por capas de la aplicación.

5.2.3.3. SERVIDOR DE BASE DE DATOS

MYSQL

Escogeremos MySQL, por tratarse de uno de los más utilizados en proyectos programados con el lenguaje de programación PHP.

Las principales características de MySQL son:

- Multiplataforma. Funciona en Windows, Linux, Mac OS X, OpenBSD, Solaris, SunOS, etc.
- Gran velocidad en la ejecución de operaciones.
- Ligera, poco consumo de recursos del equipo.
- Excelente integración con PHP.
- Conectividad segura.
- Búsqueda e indexación de campos de texto.
- Creación de usuarios con diferentes niveles de permisos, pueden limitar el acceso a tablas, bases de datos y operaciones que puedan realizar.

5.2.3.4. FRAMEWORK WEB CAKEPHP

CAKEPHP

Hemos escogido el Framework CakePHP, por extraer mayor puntuación en la votación realizada y porque entre las características más destacables se incluyen las siguientes:

- Utiliza una arquitectura basada en el patrón de diseño de software Modelo Vista Controlador (MVC) y orientada a objetos.
- Dispone de una comunidad activa de usuarios.
- Licencia flexible.
- Compatible con las versiones de lenguaje de programación PHP4 y PHP5.
- Realiza operaciones básicas en Base de Datos.
- Permite acceder a la aplicación a través de URLs amigables y configurables.
- Incorpora validaciones a lo largo del Framework.
- Genera plantillas de forma rápida y flexible.
- Incorpora asistentes de construcción de vistas: para la automatización de la generación de código en AJAX[8], JavaScript entre otros.
- Incorpora componentes de seguridad, manejo de sesiones y de peticiones.
- Dispone de listas de control de acceso flexibles.
- Almacenamiento en caché de las vistas.
- Trabaja en cualquier subdirectorio de un Servidor Web.

[8] Asynchronous JavaScript and XML.

5.2.3.5. OTRAS NECESIDADES TECNOLÓGICAS

CSS

CSS (Cascading StyleSheet) es un lenguaje de estilos.

Es el estándar reconocido por el **W3C** y la única opción viable en cuanto al diseño de las páginas web más allá de imágenes que enriquezcan la maqueta, por supuesto. Se ha utilizado cake.generic.css que ofrece el Framework CakePhp.

JAVASCRIPT (JQUERY)

JavaScript es un lenguaje desarrollado inicialmente por Netscape que permite programar la lógica de la aplicación desde el lado del cliente. Es enormemente útil y en los últimos años es el responsable del cambio de concepto de la web que la ha acercado a las aplicaciones de escritorio tradicionales.
JQuery es un Framework que permite el uso intensivo y sencillo de **JavaScript**, así como el uso de efectos muy interesantes que hacen de la aplicación web algo vivo. La capacidad de utilizar **AJAX** tecnología que permite enviar peticiones al servidor sin cambiar de página ha permitido que en los últimos tiempos el comportamiento habitual de una página web, esto es, efectuar una opción y esperar una respuesta por parte del servidor que implicaba necesariamente un cambio de página, cambie totalmente.

Las desventajas son:

- Código visible por cualquier usuario.
- El código debe descargarse completamente.

PHPEXCEL

PhpExcel, proporciona un conjunto de clases para el lenguaje de programación PHP, que le permiten escribir y leer en diferentes formatos de archivos de hojas de cálculo, como Excel (BIFF). Xls, Excel 2007 (OfficeOpenXML). Xlsx, CSV, Libre / OpenOffice Calc. Ods , Gnumeric, PDF, HTM. Este proyecto se basa en el estándar OpenXML de Microsoft y PHP.

5.3. HERRAMIENTAS UTILIZADAS

Para la realización del portal web, se han empleado diferentes programas dependiendo del trabajo que se tuviera que realizar. A continuación se muestra una relación y una explicación de cada uno de ellos, la parte de instalación y configuración aparecen en el ANEXO- 2 TOOL USED

XAMPP

Se ha utilizado XAMPP para desarrollar la aplicación del Portal Web del proyecto, una aplicación que recoge el servidor HTTP Apache, intérprete de scripts PHP y gestor de bases de datos MySQL. También dispone de un servidor FTP.

Esta herramienta facilita la instalación de todos los elementos necesarios para instalar un Servidor Web de forma ágil y rápida y dispone de un Panel de Control.
Los servicios necesarios para la realización del Portal Web son:

- Servidor Web HTTP APACHE.
- Gestor de bases de datos MySQL.
- Intérprete scripts PHP.
- Administrador de Bases de Datos phpMyAdmin.

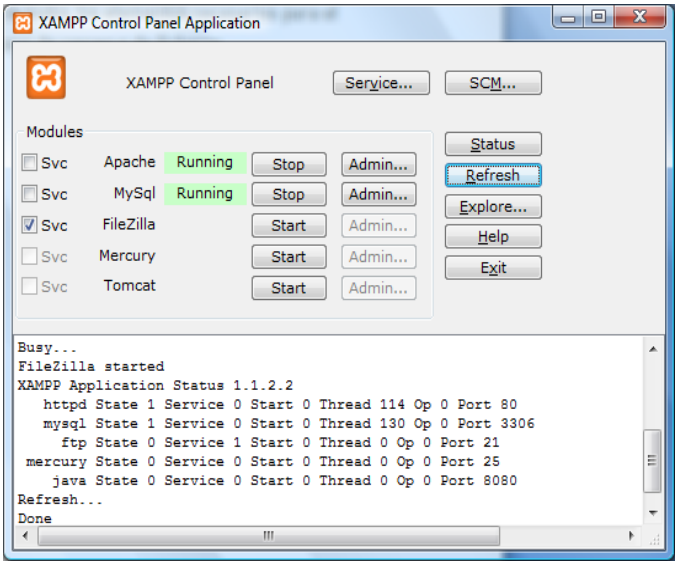


Ilustración 14 –Xampp - Panel de Control

NOTEPAD++

Notepad++ se distribuye bajo los términos de la Licencia Pública General de GNU. Se ha utilizado el programa **Notepad++** como editor de texto y de código fuente libre con soporte para varios lenguajes de programación. Incluye opciones que son útiles para usuarios avanzados como desarrolladores y programadores.

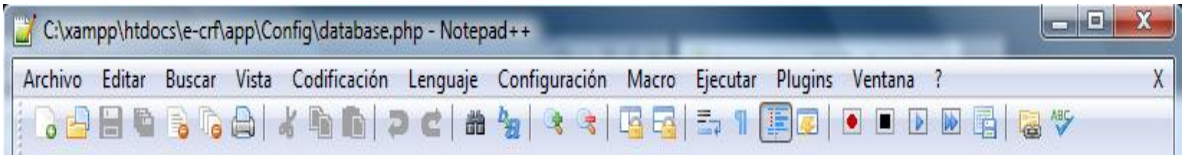


Ilustración 15 – Panel del Notepad ++

GANTT PROJECT

Gantt Project es un programa gratuito para hacer la gestión y administración de recursos y tareas a realizar en el proyecto. Este programa permite especificar y volver a planificar qué recursos se ocupan de hacer las tareas asignadas por el Analista.

MYSQL WORKBENCH VISUAL DATABASE DESIGNER 5.1

Esta herramienta permite modelar diagramas de entidad y relación para bases de datos MySQL. Se puede utilizar para diseñar el esquema de una base de datos nueva, documentar una ya existente o realizar una migración compleja. La aplicación elabora una representación visual de las tablas, vistas, procedimientos almacenados y claves foráneas de la base de datos. Además, es capaz de sincronizar el modelo en desarrollo con la base de datos real, ingeniería inversa para importar el esquema de una base de datos ya existente la cual haya sido guardada o con copia de seguridad con MySQL Administrator.

- Versiones disponibles para Sistemas Operativos Windows y Linux.

5.4. PREPARACIÓN DEL ENTORNO DE TRABAJO

Se describe con brevedad, los puntos a seguir para preparar el entorno de trabajo para desarrollar el proyecto. En este capítulo se explicará cómo se prepara y configura el Servidor Web, el Intérprete de PHP y la Base de Datos MySQL.
El ordenador utilizado durante la realización del proyecto tiene las siguientes características:

Tipo de Sistema Operativo	**Sistema Operativo Windows de 32 bits**
Procesador	Intel(R) Core(TM) 2 Duo CPD T8300 @ 2,40GHz
RAM	2,00GB

Ilustración 16 -Características Servidor de pruebas

Estas prestaciones son suficientes para la creación de la plataforma web, la instalación se realizara mediante XAMPP.
Después de llevar a cabo la instalación, configuraremos algunos parámetros para acabar de preparar el equipo. Definiremos el directorio raíz del servidor web, donde guardaremos los archivos que compondrán la página. Para ello modificaremos el archivo *httpd.conf*. En el gestor phpMyAdmin configuraremos una cuenta de usuario con privilegios y permisos suficientes para acceder a la base de datos. Y realizarnos la conexión con la base de datos.

El entorno anterior será el entorno de pruebas o desarrollo ya que disponemos de un Servidor cedido por la Universidad para alojar la aplicación web.

El servidor Holmes es una maquina virtual KVM residente en el Servidor Vardomo.
Máquina hotel: Vardomo (DELL PowerEdge R710) 2 x Intel Xeon E5506.

Este servidor dispone de las siguientes características:

Tipo de Sistema Operativo	**Sistema Operativo Linux**
Procesador	(2.13GHz, 4M Cache, 4.86 GT/s QPI), 800MHz Max Memory
RAM	12GB

Ilustración 17-Características Servidor de Producción

5.5. ESTRUCTURA DE LA APLICACIÓN

La estructura de la aplicación viene condicionada por el patrón de diseño Modelo Vista Controlador (MVC) del Framework CakePHP que aísla la lógica que implica la interface de usuario, obteniendo como resultado una aplicación donde un cambio en la apariencia visual o cambio en la lógica no afecta.

Framework CakePHP aplica el patrón y separa la aplicación en tres partes bien diferenciadas:

- Los Modelo**:** Es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos.

- Las Vistas: Presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.

- Los Controladores**:** Responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

También incluye otras clases y objetos que hacen que el desarrollo en Modelo Vista Controlador (MVC) sea más rápido y agradable. Los Componentes [Components], Comportamientos [Behaviors], y Ayudantes [Helpers] son clases que proporcionan extensibilidad y reusabilidad; agregan rápidamente funcionalidad a las clases base MVC de las aplicaciones.

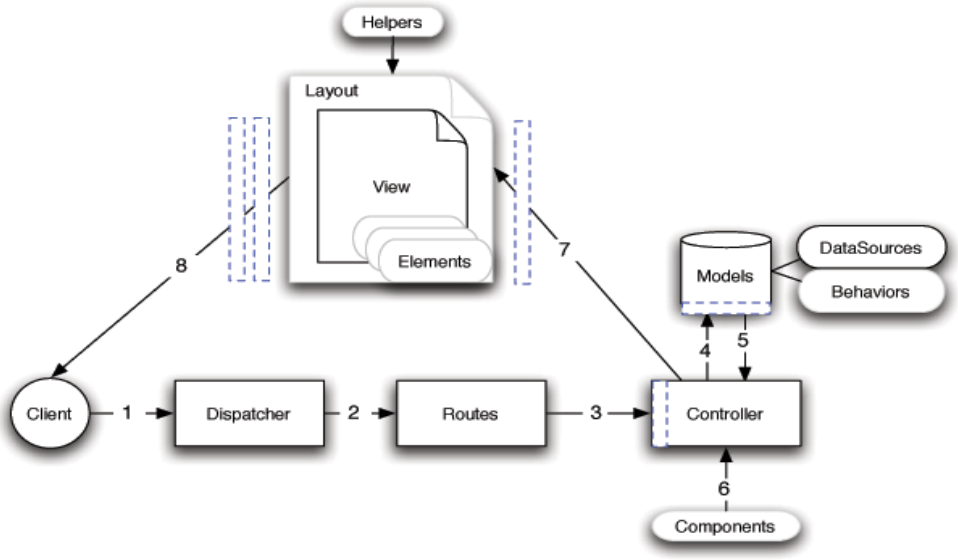


Ilustración 18 –Llamada con CakePHP

Directorios

A continuación se muestra el árbol de directorios de CakePHP

/ecrf

/config	//Archivo de configuración.
/controllers	**//Controladores**
/libs	//Librerías opcionales
/locale	//Archivos
/models	**//Modelos representativos de los datos**
/plugins	//Módulos para la aplicación
/tests	//Archivos de test
/tmp	//Ficheros temporales usados en la ejecución
/vendors	//Clases y scripts
/views	**//Archivos de redirección de vistes (.ctp)**
/webroot	//CSS, imágenes, archivos Java script,...
index.php	

5.6. DISEÑO GRÁFICO DE LA WEB

El diseño gráfico o interfaz grafica de las pantallas que hace referencia a la apariencia que tendrá la aplicación, se basa en el diseño que dispone por defecto el Framework CakePHP. Este diseño contiene el código de presentación de la vista. Cualquier cosa que queramos ver en todas las vistas las hemos situado en el *Layout*. Estos ficheros de diseño se encuentran en la ruta */app/views/layouts* de nuestra aplicación.
El diseño por defecto que ofrece CakePHP */app/views/layouts/*default.ctp puede sustituirse creando un nuevo diseño.

Se pueden crear tantos diseños como se desee, simplemente hay que colócalos en el directorio app/views/layouts e intercambiadlos dentro de las acciones del controlador.

El Framework CakePHP lleva incorporado una hoja de estilos situada dentro de la ruta *ecrf/app/webroot/css* de nuestra aplicación, es una hoja de estilos muy completa, con varios estilos para los campos Input, Select, etc.

Como hemos comentado antes no es una prioridad de este proyecto por este motivo se ha utilizado el diseño que trae por defecto el Framework CakePHP *default.ctp*.

5.7. DISEÑO DE LA ESTRUCTURA DE LA APLICACIÓN

A continuación mostramos un ejemplo del diseño de una sección.

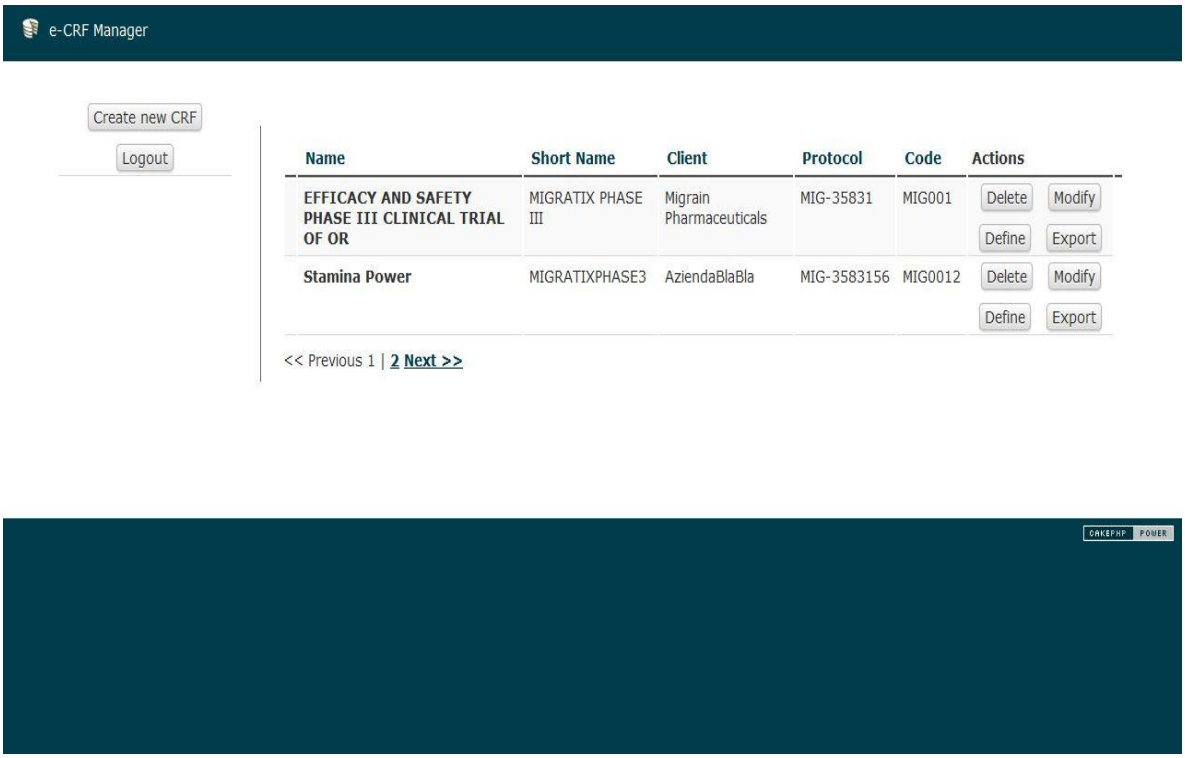


Ilustración 19- Diseño de la estructura de la aplicación I

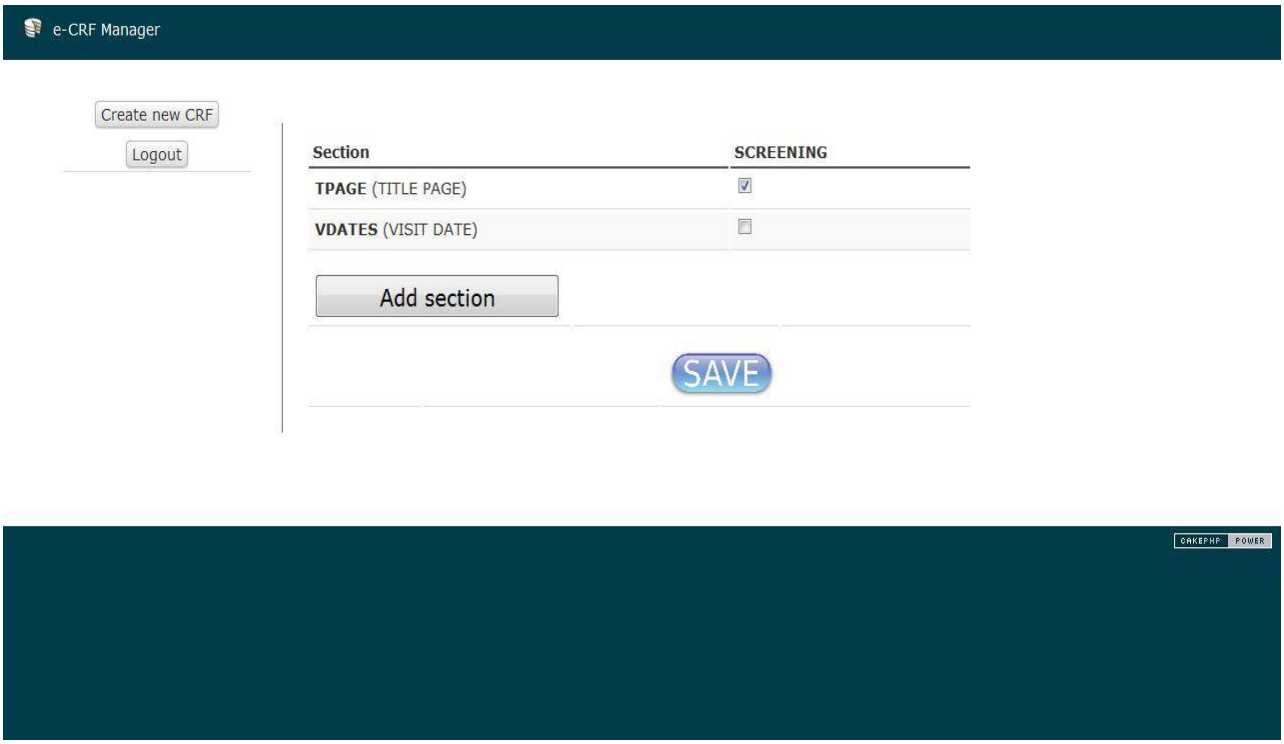


Ilustración 20- Diseño de la estructura de la aplicación II

Ilustración 21 -Diseño de la estructura de la aplicación III

5.8. ESPECIFICACIÓN Y DISEÑO DE LOS DATOS

Para mostrar las especificaciones del proyecto, se realiza el modelo conceptual de la aplicación y sus restricciones contextuales y se explica las clases que se utilizarán y los casos de uso.

5.8.1. MODELO CONCEPTUAL

Un modelo se representa en la base de datos como una tabla. CakePHP lee esta información de la estructura de la tabla automáticamente de la Base de Datos y la utiliza como base del modelo. Los ficheros de un modelo de CakePHP no incluyen ningún tipo de información sobre los campos que contiene. Toda la información se establece automáticamente con el enlace que tiene con la tabla de Base de Datos.

El esquema del modelo conceptual y la creación de las tablas de nuestro sistema son los siguientes.

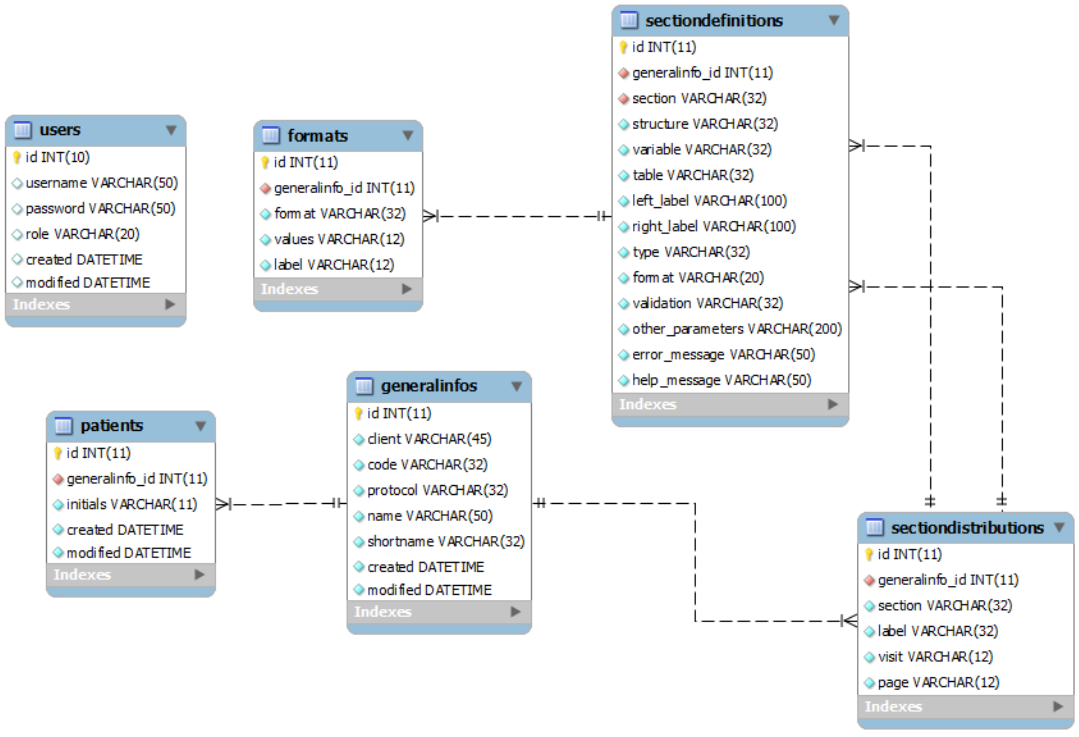


Ilustración 22 – Modelo Conceptual

users

Columna	Tipo	Nulo	Predeterminado
id	int(10)	No	
username	varchar(50)	Sí	*NULL*
password	varchar(50)	Sí	*NULL*
role	varchar(20)	Sí	*NULL*
created	datetime	Sí	*NULL*
modified	datetime	Sí	*NULL*

Tabla 1- Tabla Users

generalinfos

Columna	Tipo	Nulo
id	int(11)	No
client	varchar(45)	No
code	varchar(32)	No
protocol	varchar(32)	No
name	varchar(50)	No
shortname	varchar(32)	No
created	datetime	No
modified	datetime	No

Tabla 2 -Tabla GeneralInfos

Sectiondistributions

Columna	Tipo	Nulo	Enlaces a
id	int(11)	No	
generalinfo_id	int(11)	No	generalinfos -> id
section	varchar(32)	No	
label	varchar(32)	No	
visit	varchar(12)	No	
page	varchar(12)	No	

Tabla 3- Sectiondistributions

Sectiondefinitions

Columna	Tipo	Nulo	Enlaces a
id	int(11)	No	
generalinfo_id	int(11)	No	sectiondistributions -> generalinfo_id
section	varchar(32)	No	sectiondistributions -> section
structure	varchar(32)	No	
variable	varchar(32)	No	
table	varchar(32)	No	

left_label	varchar(100)	No	
right_label	varchar(100)	No	
type	varchar(32)	No	
format	varchar(20)	No	
validation	varchar(32)	No	
other_parameters	varchar(200)	No	
error_message	varchar(50)	No	
help_message	varchar(50)	No	

Tabla 4- Tabla Sectiondefinitions

Estructura de tabla para la tabla formats

Columna	Tipo	Nulo	Enlaces a
id	int(11)	No	
generalinfo_id	int(11)	No	sectiondefinitions -> generalinfo_id
format	varchar(32)	No	
values	varchar(12)	No	
label	varchar(12)	No	

Tabla 5 - Tabla Formats

patients

Columna	Tipo	Nulo	Enlaces a
id	int(11)	No	
generalinfo_id	int(11)	No	generalinfos -> id
initials	varchar(11)	No	
created	datetime	No	
modified	datetime	No	

Tabla 6- Tabla Patients

Los convenios de CakePHP para las tablas son los siguientes:

- Los nombres de las tablas se definen en plural y minúsculas, mientras que la clase modelo se define en singular. Para la clase Patient, se define la tabla patients.

- Todas las tablas para las que se defina una clase modelo, necesita la clave primaria el atributo id. Esta clave debe tener la propiedad de ser un campo auto- incremental.

- Si una tabla posee los atributos *created* y *modified*, su valor se actualiza automáticamente por CakePHP cuando se crea o modifica.

Model(modelo)

Cada tabla debe tener una clase modelo de igual nombre que la tabla, pero en

singular, guardada en un archivo también con el mismo nombre de la tabla, pero en singular, con extensión '.php'. Dicho fichero deberá almacenarse en la carpeta '*[…]/ecrf/app/ models/*'.

Controller (controlador)

El nombre de la clase controlador se forma poniendo el nombre de la tabla en plural, seguido de '*Controller*'. Esta clase deberá guardarse en un fichero dentro de la carpeta '*[…]/ecrf/app/controllers/*'.

Views (vistas)

Para crear las vistas de las tablas de la base de datos, hay que crear una carpeta por cada tabla dentro de la carpeta '*[…] /ecrf/app/views/*'. Dentro de cada una de las carpetas creadas para las vistas se creará un fichero para la vista de edición, de nombre '*edit.ctp*', otro para la de inserción, '*add.ctp*', otro para el listado de registros '*inde.ctp*' y otro para cada registro '*view.ctp*'.

Sobre el código de creación en SQL de las tablas destacamos:

- El campo id debe ser PRIMARY KEY y debe tener el atributo auto_increment.
- Los campos de texto deben usar el collate utf8_unicode, esto indica que el texto es UTF-8 y que las búsquedas sobre este campo no distinguen entre mayúsculas y minúsculas (case insensitive).
- Se indexan los campos relacionados. Por ejemplo generalinfo_id en la tabla sectiondistributions.
- El motor de las tablas debe ser InnoDB. Permite transacciones.
- Las tablas de clases asociativos no deben tener el atributo id, únicamente tienen las dos claves foráneas.

Reglas de validación

CakePHP, posee la característica de validar datos de formularios, las reglas de validación que define el Cakephp evitan problemas y son indispensables para garantizar la integridad de los datos.

- **alphaNumericos**: El campo solo puede contener letras y números.
- **boolean:** El campo a de ser un valor booleano.los valores validos son true o false o 1 o 0.
- **date: Valida** que el campo sea una fecha correcta. Tiene un parámetro para indicar formato de fecha.
- **Decimal**: Asegura que el campo contente un valor decimal. Tiene in parámetro para solicitar el número exacto de decimales.
- **inList**: la regla asegura que el valor del campo es un valor de una lista que se pasa por parámetro.
- **Numeric:** el valor a de ser un valor numérico valido.
- **notEmpty:** regla básica para asegurar que el campo no este vacío.

  A parte cada regla de validación permite indicar si el campo es:
  - **Requiered:** el campo tiene que estar presente.
  - **allowEmpty**: el campo permite valores nulos o vacios independientemente de la regla de validación.

Para hacer esta validación en CakePHP, tendremos que crear las validaciones en las vistas.

Atributo Uses

El atributo Uses permite que el Controller pueda utilizar un Modelo con un nombre deferente al modelo por defecto. Él atributo también permite declarar un Controller que no utilice ningún Modelo o que utilice más de uno.

Helpers

El atributo helper permite añadir a un Controller el uso de Helpers que utilizan a las vistas que controla. Un helper es una clase que encapsula tareas que pueden ser compartidas entre diferentes vistas u ayuda a generar código de la capa de presentación.

Components

El atributo Components permite añadir Components al Controller para ayudar en la lógica de las acciones. Es una clase que encapsula tareas lógicas que pueden ser comparadas entre diferentes Controllers.

Views

Les vistes forman la capa de presentación, donde se interactúa con el usuario de la aplicación web. Las vistas son ficheros de extensión *.ctp que presentan el comportamiento de plantillas.

El esquema conceptual correspondiente al dominio de la aplicación web está representado en el ANEXO- 3 CONCEPTUAL MODELS

5.8.2. DIAGRAMA DE CLASES

El diagrama de clases describe la estructura del sistema mostrando las clases, atributos y las relaciones entre ellas. Podemos ver el diagrama a continuación, también se muestra en el ANEXO 4- CLASS DIAGRAM

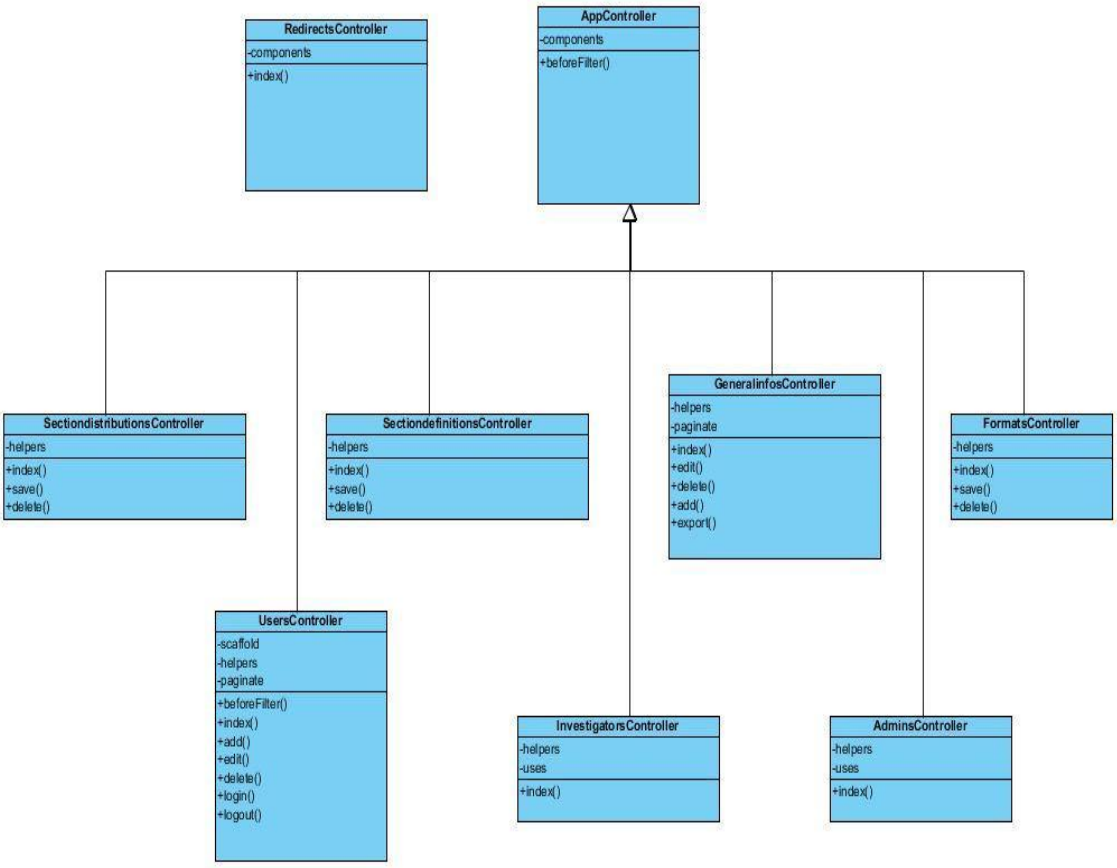


Ilustración 24 –Diagrama de Clases

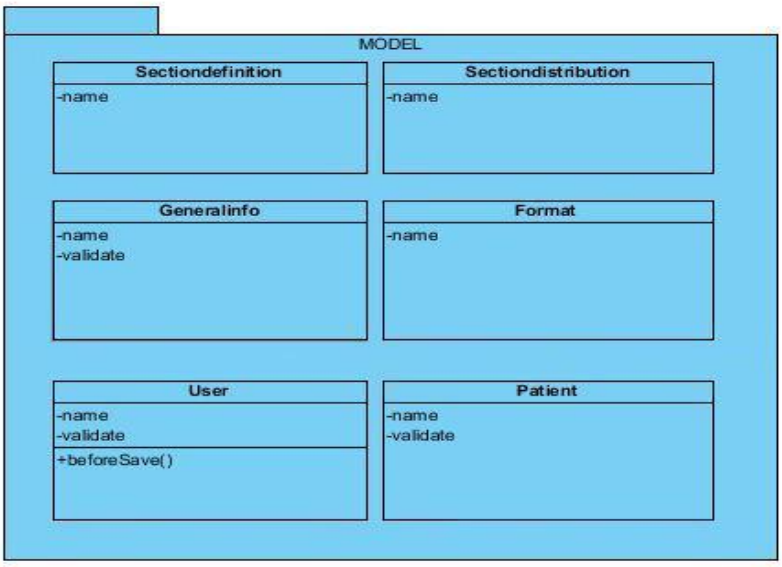


Ilustración 23 –Diagrama de Clases II

5.8.3. MODELO DE CASOS DE USO.

Una vez explicado el modelo conceptual, el siguiente paso en la especificación será elaborar el modelo de casos de uso. A diferencia del modelo conceptual, el modelo de casos de uso identifica las funcionalidades que debe tener el sistema y asocia a cada actor las funciones que deberá realizar.

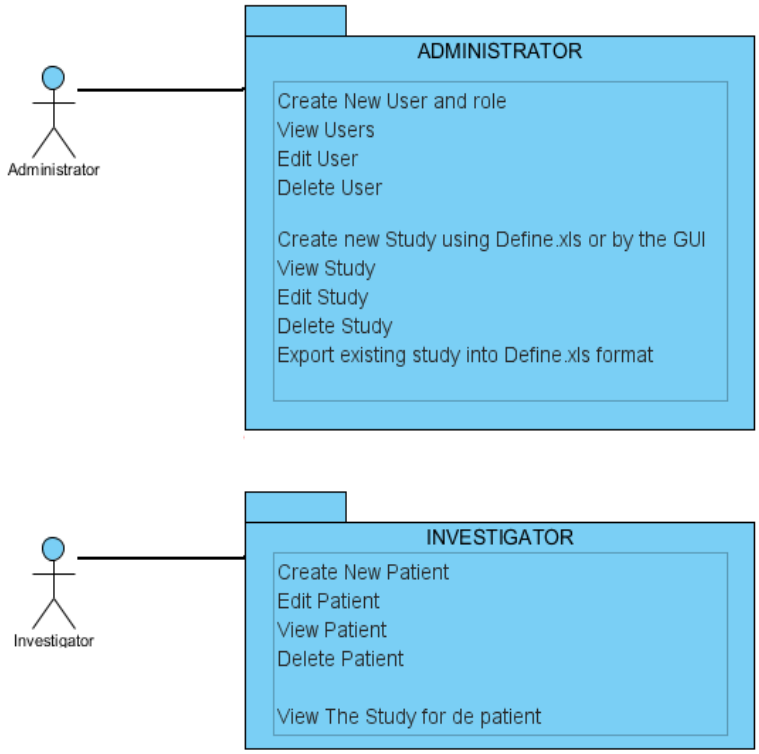


Ilustración 25 - Modelo de Caso de Uso

El diagrama de casos de uso describe como los actores interactúan con el sistema, describiendo en cada caso de uso la secuencia de eventos. En la especificación de un caso de uso se pueden definir muchas propiedades o atributos pero en nuestro caso, especificaremos las siguientes:

Caso de uso	**Nombre del caso de uso que especifiquemos**
Actores	Agentes
Curso típico de eventos	Descripción de la interacción

Ilustración 26 - Especificación de un caso de uso

Las secuencias de eventos de cada caso de uso son las siguientes:

Caso de uso: Crear nuevo usuario **Actor: Administrador**	
Acción del Administrador	**Respuesta del Sistema**
1. El administrador crea un nuevo usuario proporcionando los datos del nuevo usuario	2. Verifica que el usuario no exista.
	3a.Si existe genera error. 3b.Si no existe guarda los datos del nuevo usuario.

Ilustración 27 -Caso de Uso: Crear usuario

Caso de uso: Lista de usuarios **Actor: Administrador**	
Acción del Administrador	**Respuesta del Sistema**
1. El administrador solicita ver la lista de usuarios.	2. Muestra la lista usuarios.

Ilustración 28 -Caso de Uso: Listar de usuarios

Caso de uso: Modifica usuarios **Actor: Administrador**	
Acción del Administrador	**Respuesta del Sistema**
1. El administrador crea un nuevo usuario proporcionando los datos del nuevo usuario	2. Verifica que el usuario no exista.
	3a.Si existe genera error. 3b.Si no existe guarda los datos del nuevo usuario.

Ilustración 29 -Caso de Uso: Modifica Usuario

Caso de uso: Elimina usuarios **Actor: Administrador**	
Acción del Administrador	**Respuesta del Sistema**
1. El administrador elimina el usuario seleccionado.	2. Muestra error de confirmación conforme el usuario ha sido eliminado.

Ilustración 30 - Caso de Uso: Elimina usuario

Caso de uso: Crear nuevo estudio Actor: Administrador	
Acción del Administrador	**Respuesta del Sistema**
1. El administrador crea un nuevo estudio seleccionando de una ruta la hoja de cálculo EXCEL.	2. Crea el estudio.

Ilustración 31 - Caso de Uso: Crear estudio

Caso de uso: Listar estudios Actor: Administrador	
Acción del Administrador	**Respuesta del Sistema**
1. El administrador solicita ver el estudio.	2. Muestra la los datos del estudio.

Ilustración 32 -Caso de Uso: Listar estudio

Caso de uso: Modifica la información general del estudio Actor: Administrador	
Acción del Administrador	**Respuesta del Sistema**
1. El administrador selecciona el estudio para modificar la información general del estudio.	2. Verifica que no exista un estudio con el mismo nombre
	3a.Si existe genera error. 3b.Si no existe guarda los datos generales del estudio.

Ilustración 33 -Caso de Uso: Modifica estudio

Caso de uso: Elimina estudio Actor: Administrador	
Acción del Administrador	**Respuesta del Sistema**
1. El administrador elimina el estudio.	2. El sistema pregunta si es ese el estudio que quieres eliminar
	3. Te confirma que el estudio se ha eliminado.

Ilustración 34 - Caso de Uso: Elimina el paciente

Caso de uso: Crear nuevo paciente Actor: Investigador	
Acción del Investigador	**Respuesta del Sistema**
1. El administrador crea un nuevo paciente proporcionando los datos del nuevo paciente	2. Verifica que el paciente no exista.
	3a.Si existe genera error. 3b.Si no existe guarda los datos del nuevo paciente.

Ilustración 35 -Caso de Uso: Crear paciente.

Caso de uso: Listar pacientes Actor: Investigador	
Acción del Investigador	**Respuesta del Sistema**
1. El administrador solicita ver la lista de pacientes.	2. Muestra la lista pacientes.

Ilustración 36 -Caso de Uso: Listar de pacientes

Caso de uso: Modifica paciente Actor: Investigador	
Acción del Investigador	**Respuesta del Sistema**
1. El administrador crea un nuevo paciente proporcionando los datos del nuevo paciente	2. Verifica que el paciente no exista.
	3a.Si existe genera error. 3b.Si no existe guarda los datos del nuevo paciente.

Ilustración 37 -Caso de Uso: Modifica paciente

Caso de uso: Elimina paciente Actor: Investigador	
Acción del Investigador	**Respuesta del Sistema**
1. El administrador elimina el paciente seleccionado.	2. Confirma que se ha eliminado.

Ilustración 38 - Caso de Uso: Elimina paciente

También encontraremos los casos de uso y las especificaciones definidos en el ANEXO 5-USE CASE MODEL

5.8.4. DIAGRAMA DE SECUENCIAS DE LOS CASOS DE USO

El diagrama de secuencias nos permitirá modelar cómo será el funcionamiento de las operaciones en el sistema y nos dará un esquema de alto nivel de cómo se debería implementar cada operación.
Se crean diagramas de secuencia de todas las operaciones del sistema, que hacen referencia a las clases.

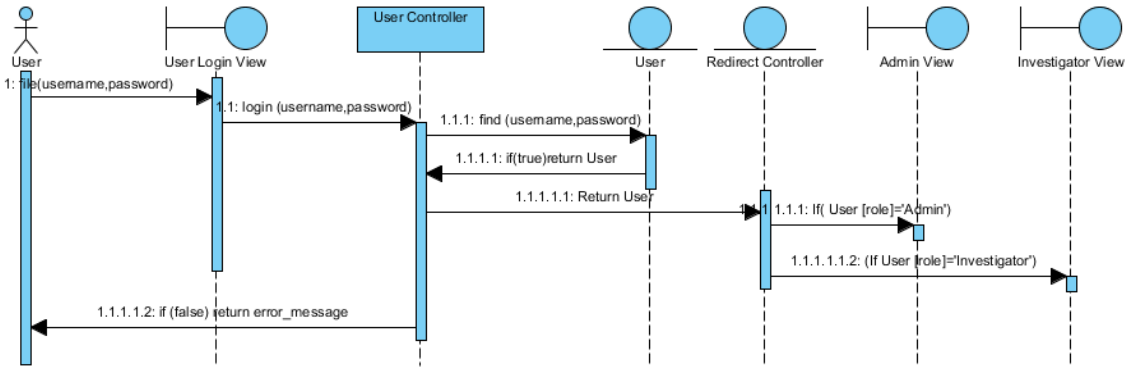


Ilustración 39- Diagrama de secuencia -Login

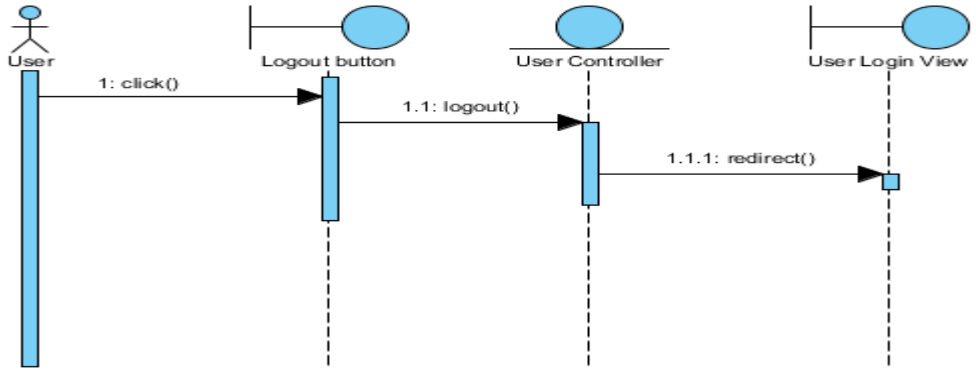


Ilustración 40- Diagrama de secuencia- Logout

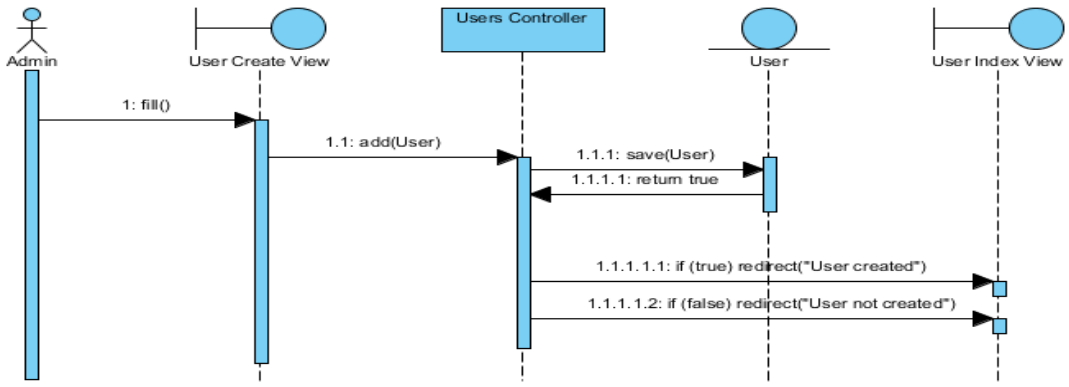


Ilustración 41- Diagrama de secuencia –Crear Usuario

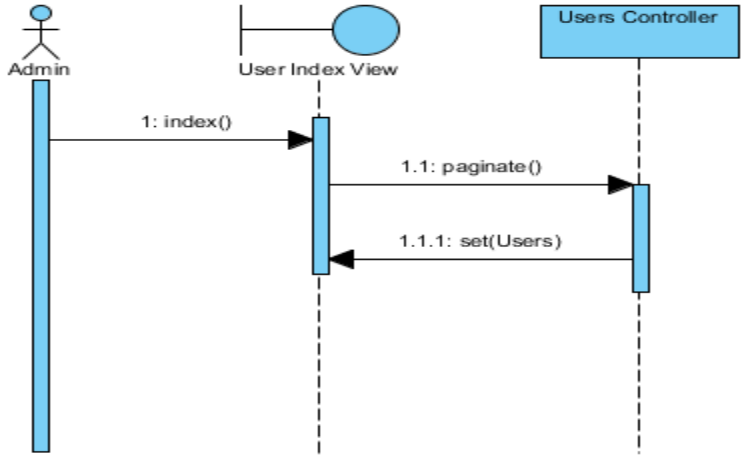


Ilustración 42- Diagrama de secuencia –Índice Usuario

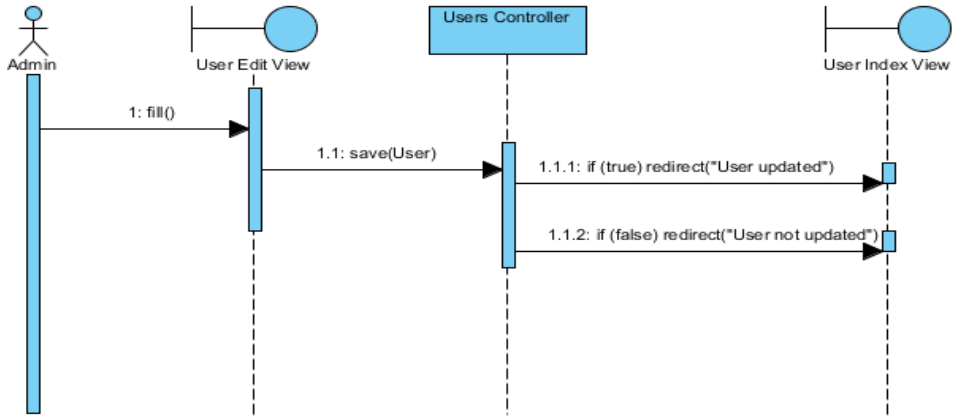


Ilustración 43-- Diagrama de secuencia Editar Usuario

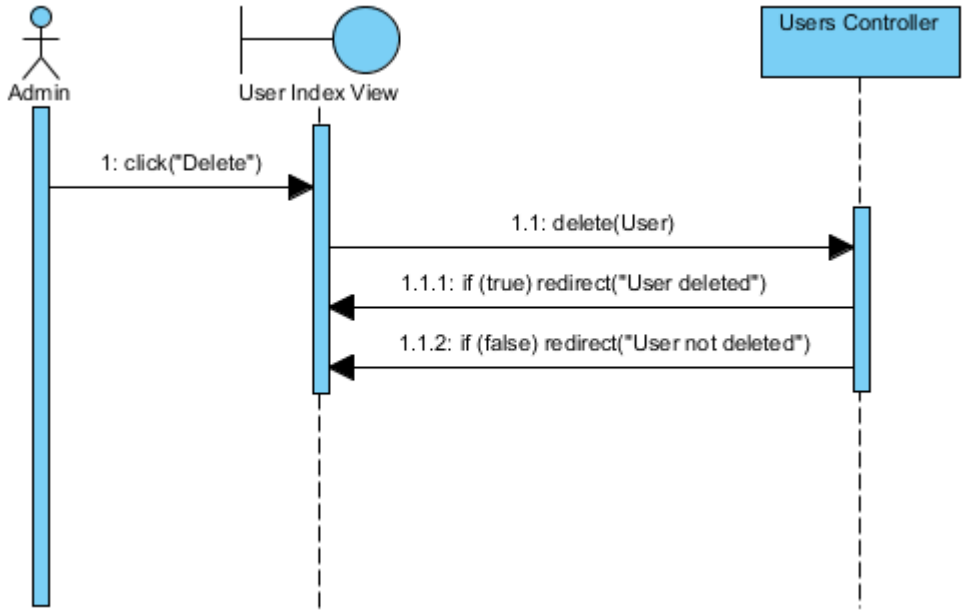


Ilustración 44 - Diagrama de secuencia- Eliminar Usuario

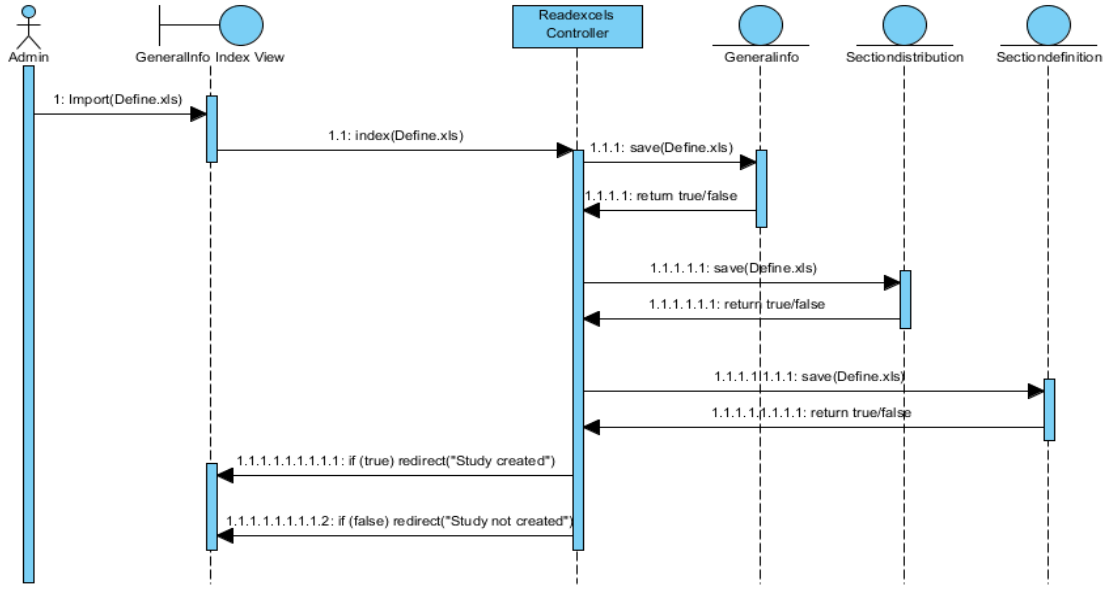


Ilustración 45- Diagrama de secuencia -Crear estudio

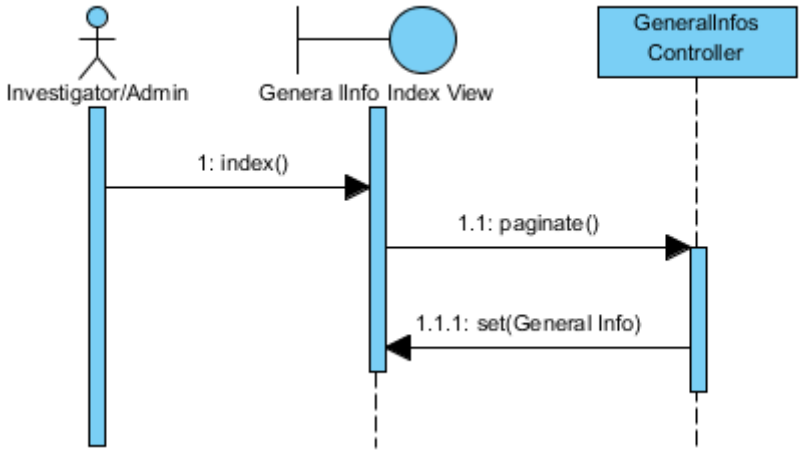


Ilustración 46 - Diagrama de secuencia- Ver Estudio

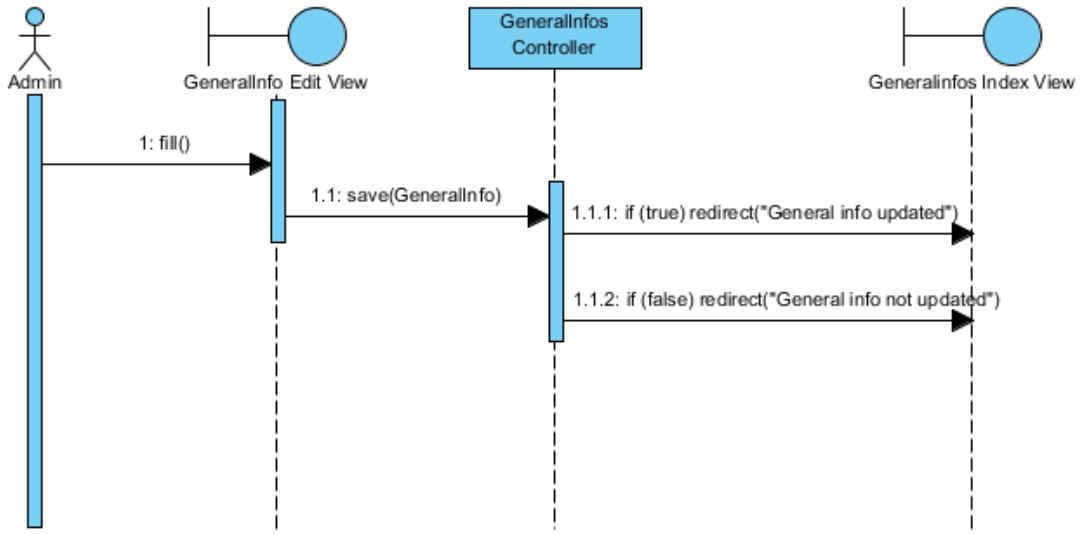


Ilustración 47- Diagrama de secuencia- Editar Estudio

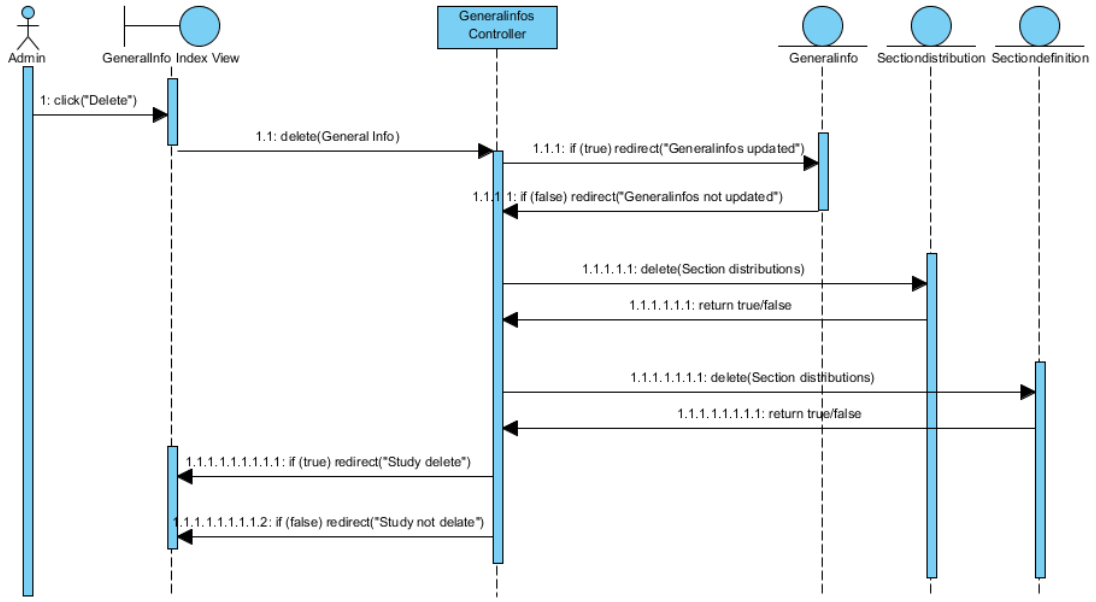


Ilustración 48- Diagrama de secuencia-Eliminar Estudio

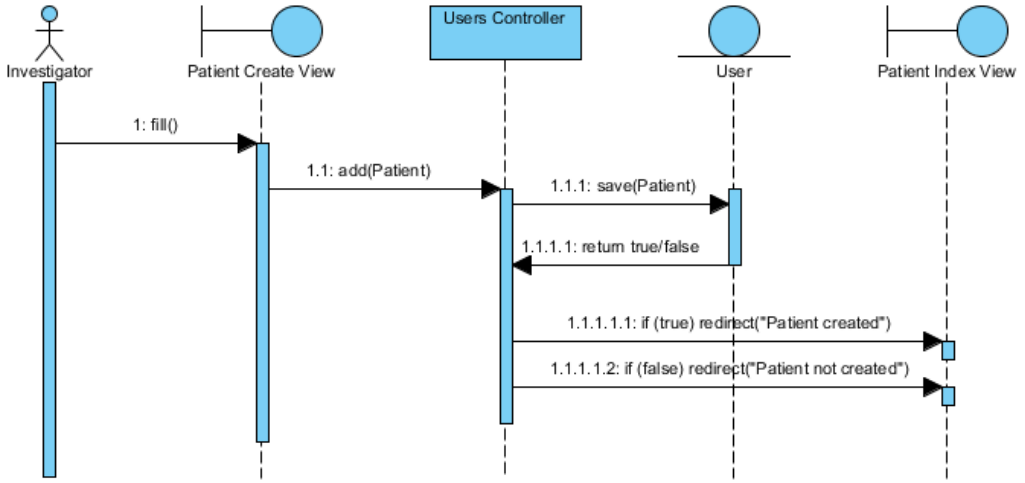


Ilustración 49 - Diagrama de secuencia- Crear Paciente

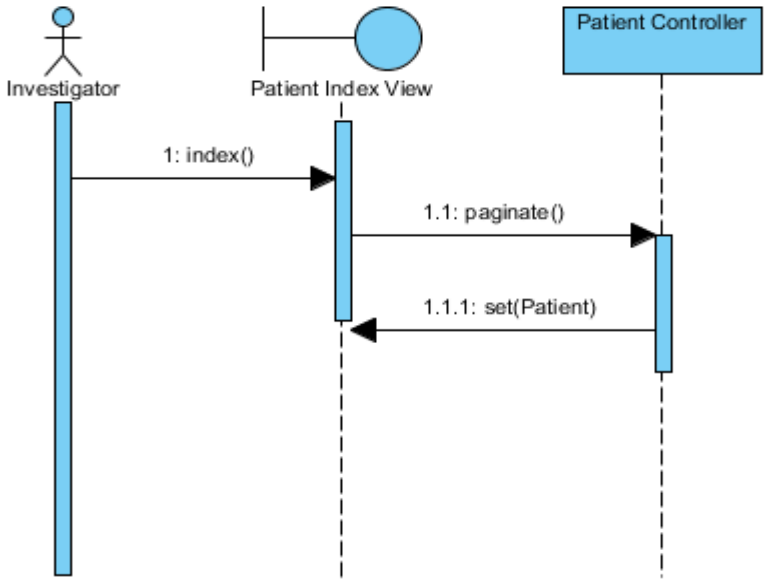


Ilustración 50- Diagrama de secuencia- Ver Paciente

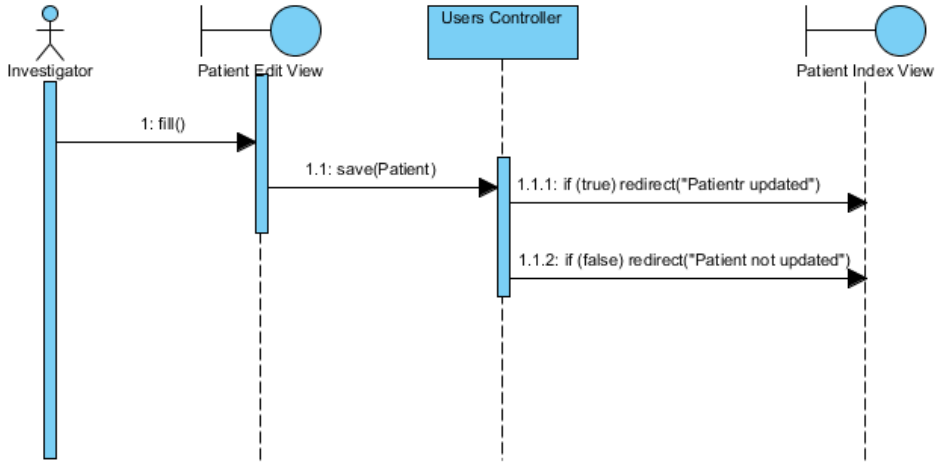


Ilustración 51 - Diagrama de secuencia- Editar Paciente

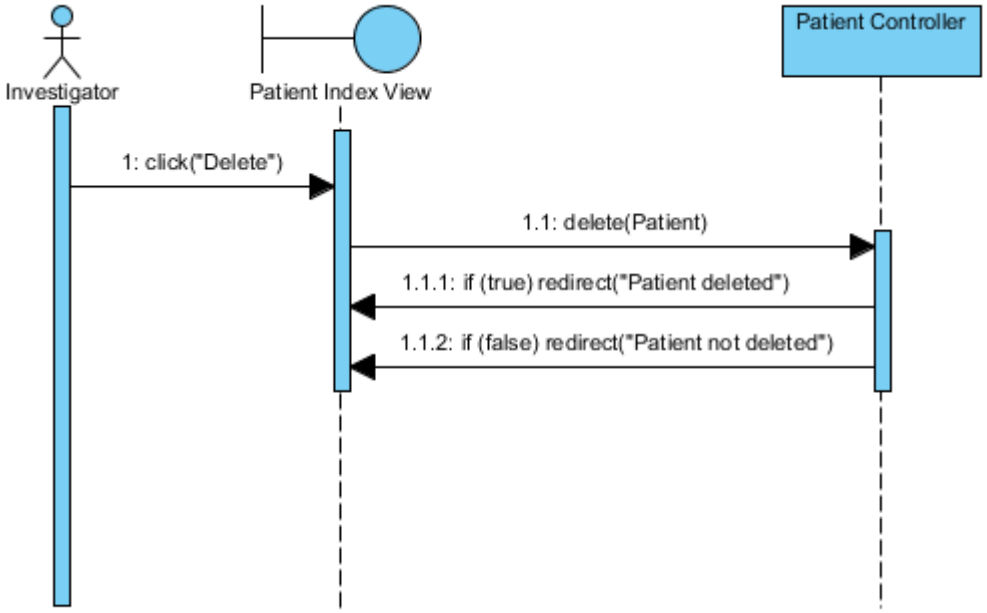


Ilustración 52 - Diagrama de secuencia -Eliminar Paciente

Se representan los diagramas de secuencia derivados de los casos de uso en el ANEXO 6- SEQUENCE DIAGRAM

5.9. DESARROLLO DE LAS DIFERENTES SECCIONES

5.9.1. SECCIÓN DE VALIDACIÓN Y ACCESO AL SISTEMA

En esta sección la validación y acceso al sistema se han realizado mediante la seguridad implantada, con ciertos parámetros de la sesión y privilegios según el rol de usuario.
CakePHP es un Framework de PHP que tiene soporte para un sistema de permisos muy complejo en base a ACL[9]. Sin embargo no todos los desarrollos web requieren de algo tan complejo. Para esos casos, con sólo usar el AuthComponent se puede manejar de manera sencilla.
Este método asume que cada usuario pertenece a un sólo grupo. Para lograrlo es necesario modificar el AppController para hacer que cargue el AuthComponent y un archivo en el que indicaremos los permisos.

En este archivo, que debemos colocar en el directorio *config* de nuestra aplicación, definimos dos cosas:

1. Los grupos permitidos en el sistema (en este caso Investigador y Administrador).

2. Por cada controlador, las acciones y el grupo que puede ejecutar dicha acción.

En el AppController se realiza todo el manejo de los permisos, de modo que los demás controladores queden libres de manejar esta lógica.

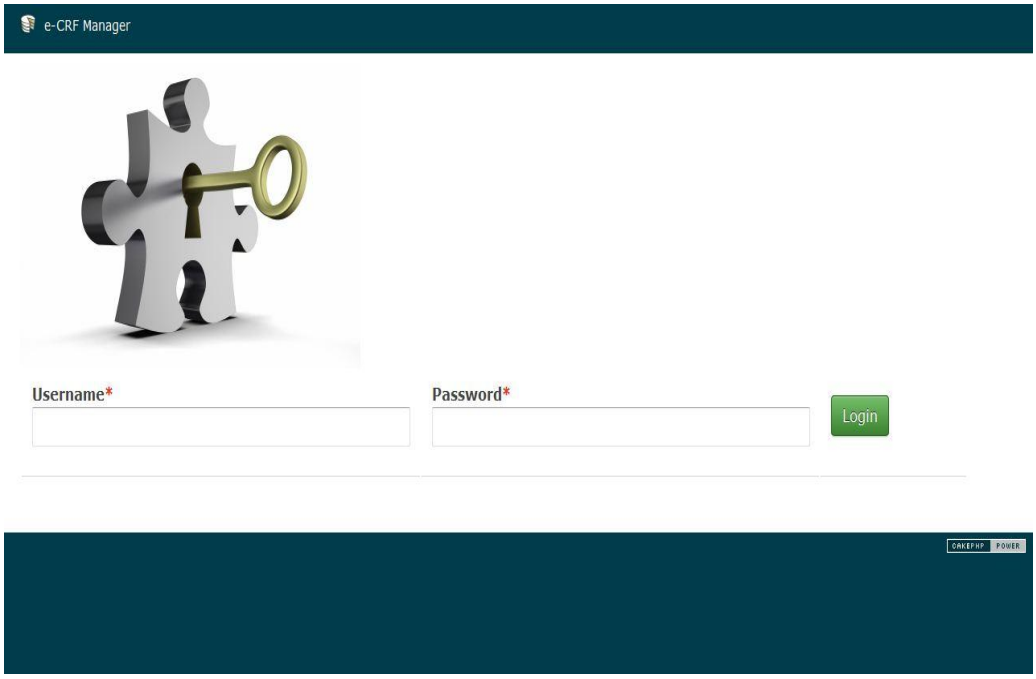


Ilustración 53- Inicio de Sesión

[9] Acl component cakephp

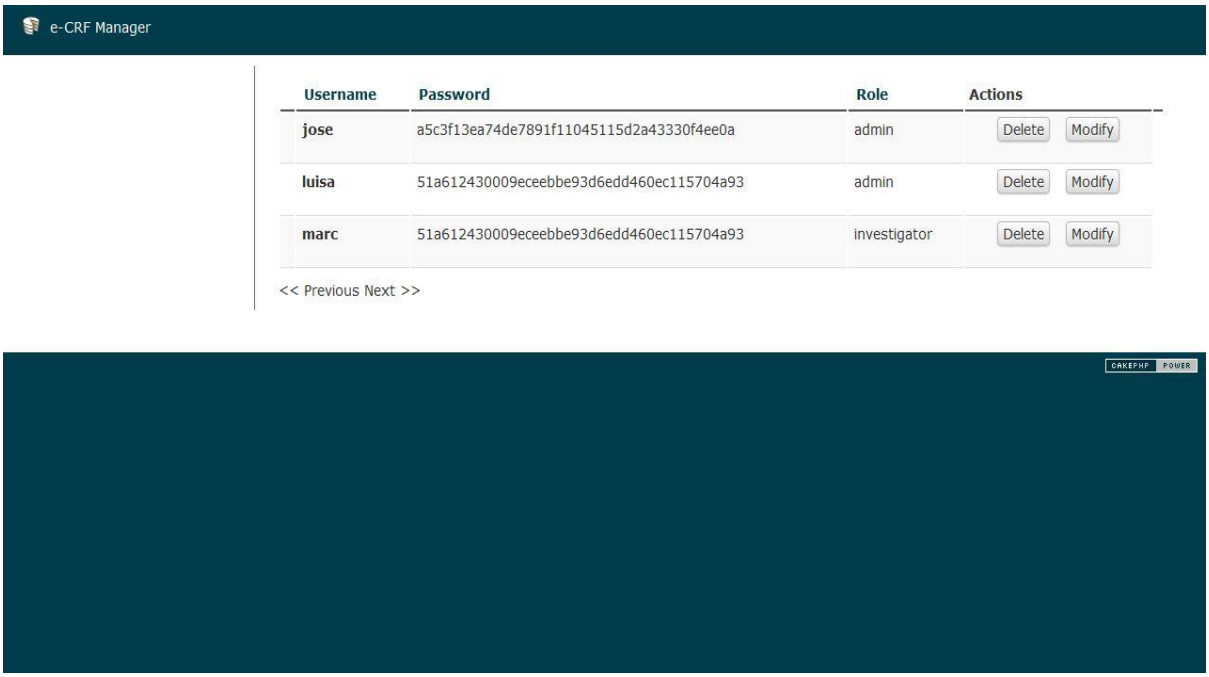


Ilustración 54 - Lista de usuarios del Sistema

5.9.2. SECCION ADMINISTRADOR Y INVESTIGADOR

Mostramos a continuación la sección grafica del Administrador.

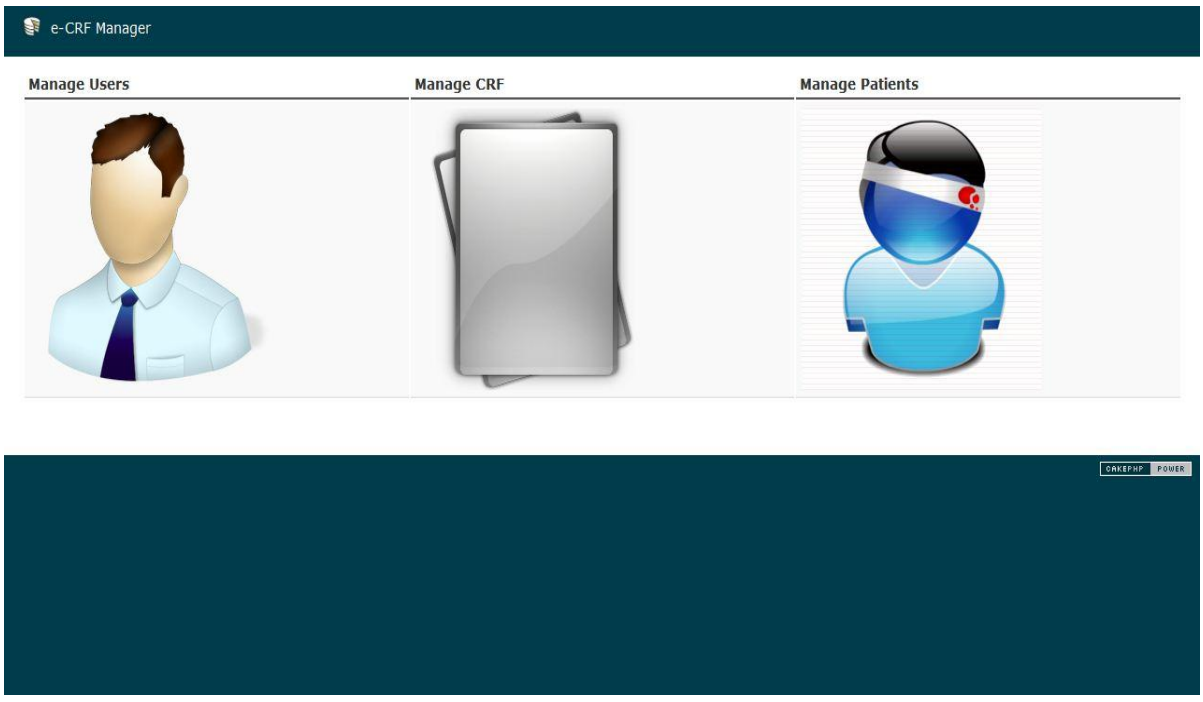


Ilustración 55-Sección Administrador

Se muestra a continuación el interface grafica de la sección del Investigador.

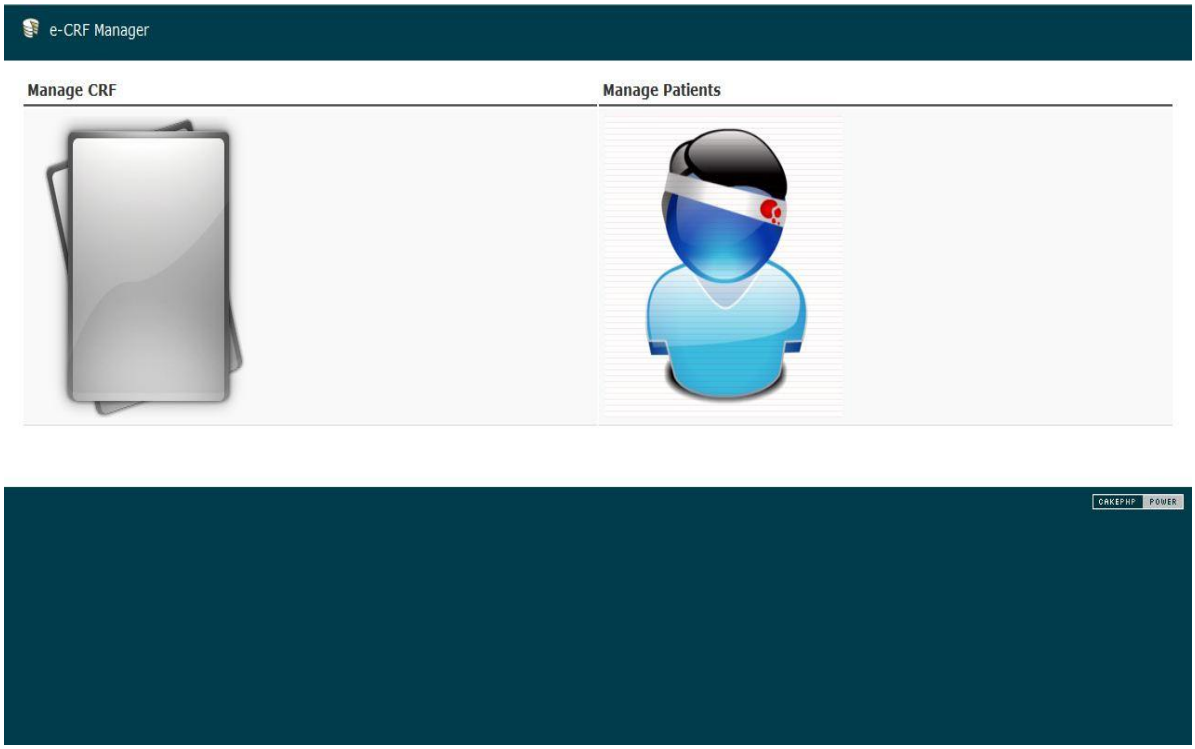


Ilustración 56- Sección Investigador

5.10. ANALISIS DE COSTES

En este apartado realizaremos la valoración económica del coste para la realización del proyecto teniendo en cuenta los recursos humanos, los costes del software y hardware necesarios para llevar a cabo el proyecto.

Para reflejar el coste de este proyecto nos basamos en toda la infraestructura de la aplicación. Podemos agrupar nuestro proyecto en:

- Valoración del coste del software: Cuantificamos el valor de los programas y de la plataforma sobre la que se mantiene el sistema.

- Coste de hardware: Coste de los ordenadores y periféricos sobre los que instalamos nuestro sistema.

- Coste del personal: Aquí valoramos el número de horas /perfil persona que consideramos que ha sido necesario, para elaborar el proyecto.

- Coste físico: Aquí valoramos el espacio en disco físico que ocupa la aplicación.

El coste de todas las aplicaciones y herramientas utilizadas es cero.

SOFTWARE	
CONCEPTO	TOTAL
XAMPP	Gratis
Editor de textos Notepad++	Gratis
GranttProject	Gratis
MySQL Workbench Visual Database Designer 5.1	Gratis
Software y Herramientas básicas cedidas Universidad	Gratis
Totales	**0€**

Ilustración 57 -Análisis de Coste - Software

En relación del proyecto, el hardware ha sido cedido por la Universidad. Se ha querido reflejar este coste. El coste del hardware se ha incluido dentro del presupuesto de los costes total del proyecto.

HARDWARE	
CONCEPTO	TOTAL
Servidor UPC	2.000€
SAI	400€
ROUTER(CONEXION 20M)	700€
ORDENADOR LOCAL	1200€
Totales	**4.300€**

Ilustración 58 -Análisis de Coste - Hardware

Como se ha tenido que aprender a utilizar las herramientas utilizadas en el proyecto, esto ha supuesto un consumo más elevado en horas dedicadas a formación.

A continuación se realiza una aproximación en el número de personas que deberían realizar el proyecto. Dividiremos la elaboración del trabajo en dos personas, el analista y el programador que también realizara las pruebas. Los precios hora para las categorías profesionales son:

Recurso	Precio-Hora por categoría
(JP)-Dirección de Proyecto/Analista:	80 €
(T)-Programador/ Verificador datos (Pruebas):	40 €

Ilustración 59 - Análisis de Coste -Precio hora por categoría

El analista hará todas las funciones de análisis de requerimientos y de la elaboración del modelo conceptual, además tiene que hacer el diseño de los diagramas de casos de uso, y el diseño de la base de datos. Deberá realizar la interfaz gráfica de los diferentes usuarios. El programador realizará las interfaces ó pantallas de la aplicación y debe hacer las pruebas del software generado. También hará la elaboración de la interfaz y la instalación de la plataforma y la puesta a punto de los servidores.

DISEÑO

Actividad	Duración	Dedicación	Perfil	Trabajo
Dirección de Proyecto	20	27%	JP	5,3
Búsqueda de soluciones	3,75	400%	T	15
Selección de plataforma	10	100%	T	10
Seguridad	6,5	200%	T	13
Documentación	2,5	300%	T	5
Totales por Perfiles				
Dirección de Proyecto/Analista (JP)				5,3
Técnico(Programador) (T)				43

Ilustración 60- Análisis de Coste – Diseño

INTEGRACIÓN

Actividad	Jornadas	Dedicación	Perfil	**Trabajo**
Dirección Proyecto	3,3	100%	JP	3,3
Documentación	12	100%	T	12
Configuración S.O	3,6	100%	T	3,6
Instalación SW	8,1	100%	T	8,1
Integración	22,5	100%	T	22,5
Pruebas	10,8	100%	T	10,8
Totales por Perfiles				
Dirección de Proyecto/Analista (JP)				3,3
Técnico(Programador) (T)				57

Ilustración 61 –Análisis de Coste – Diseño- Integración

DEMOSTRACIÓN				
Actividad	Jornadas	Dedicación	Perfil	Trabajo
Dirección de Proyecto	0,66	100%	JP	0,66
Demo	11,4	100%	T	11,4
Totales por Perfiles				
	(Dirección de Proyecto/Analista (JP)			0,66
	Técnico(Programador) (T)			11,4

Ilustración 62- Análisis de Coste –Demostración

IMPORTE POR PERFILES						
Perfil	TOTAL	Porciento	P/Hora	Importe	Dto.	Importe
Dirección de Proyecto/Analista (JP)	9,26	7%	80	740	0%	740
Técnico(Programador) (T)	18,9	15%	40	1.512	0%	1.512
TOTAL JORNADAS	28,16			2.252		2.252

Ilustración 63- Análisis de Coste –Importe por Perfil

Para hacer el cálculo total del coste del proyecto, nos hemos basado en los presupuestos elaborados previamente, los cuales contemplan todos los costes de los recursos utilizados.
Tenemos como resultado que el valor de este proyecto, supone un coste final de:
Coste software + Coste hardware + Coste personal = 2.252€ + 4.300€= 6.552€

En este apartado definiremos el coste físico, el espacio físico en disco que ocupa toda la aplicación se muestra a continuación.

DATOS	
CONCEPTO	TOTAL ESPACIO EN M
ESPACIO FISICO EN DISCO	31M
Espacio Mysql Server	20M
Totales	51M

Ilustración 64 -Análisis de Coste –Datos

6. CONTINUIDAD APLICACIÓN FUTURA

El proyecto se encuentra cerrado y cumple los requisitos. Siempre pueden surgir nuevas necesidades de requisitos o mejoras en las estrategias y los métodos usados. En este punto se enumeran algunas mejoras que podría incorporar y que se consideran que darían valor añadido a las funcionalidades del la aplicación.

Existen muchas mejoras posibles y a continuación las representamos en una lista.

Se podría adjuntar la documentación del Protocolo y el documento del consentimiento informado. Y se podrían informar de los siguientes campos mostrados en la imagen como podría ser la entrada de un control de calidad. Y dar la posibilidad de adjuntar el logo del estudio.

Ilustración 65 - Continuidad de aplicación futura I

Mejoría de la interface grafica de usuario para crear secciones intermedias y visitas intermedias utilizando la tecnología JQuery. Mostramos como podría ser esta interface en la siguiente imagen.

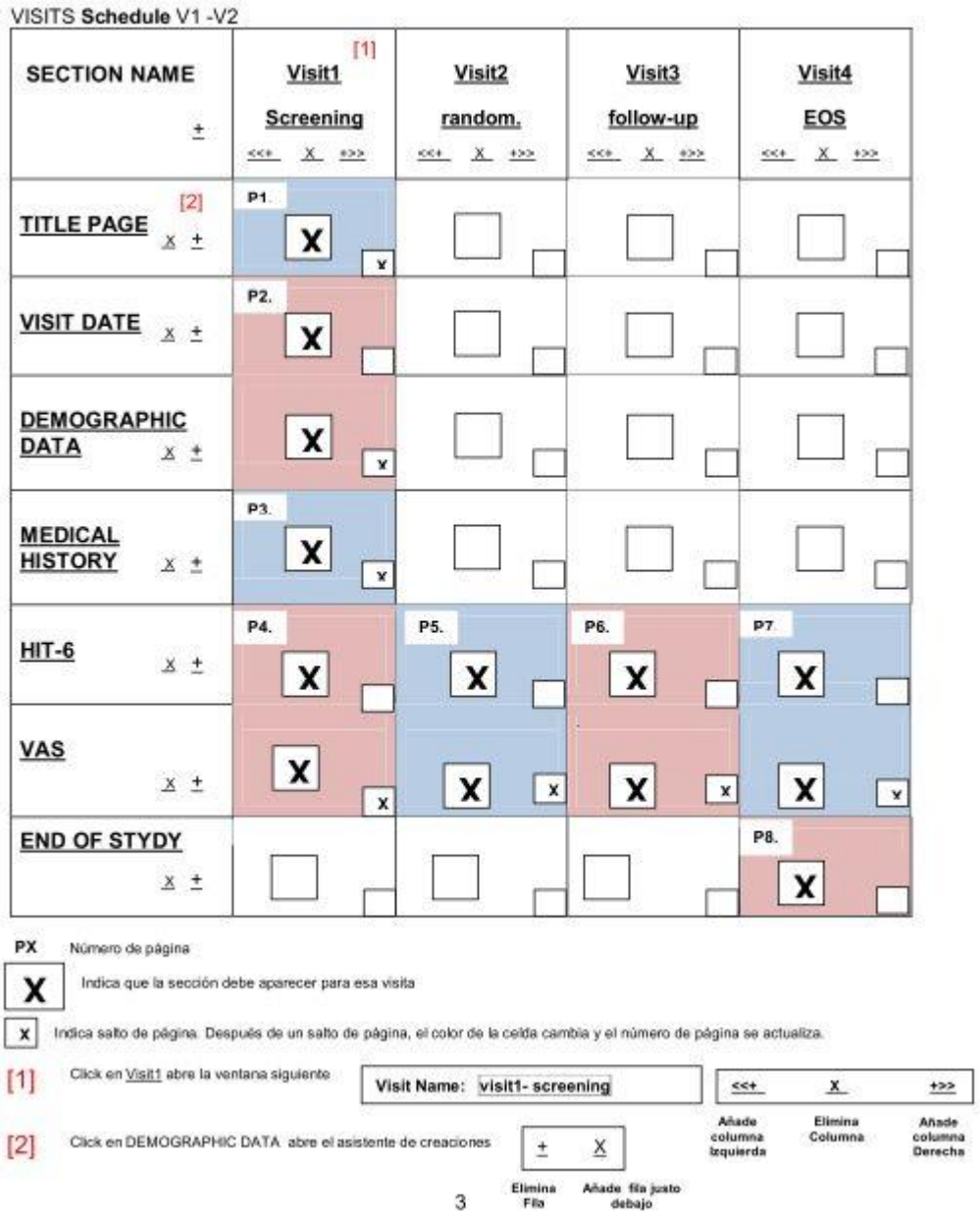


Ilustración 66 - Continuidad de aplicación futura II

Otra posible mejora sería adaptar la aplicación para que pueda ser multicentro.

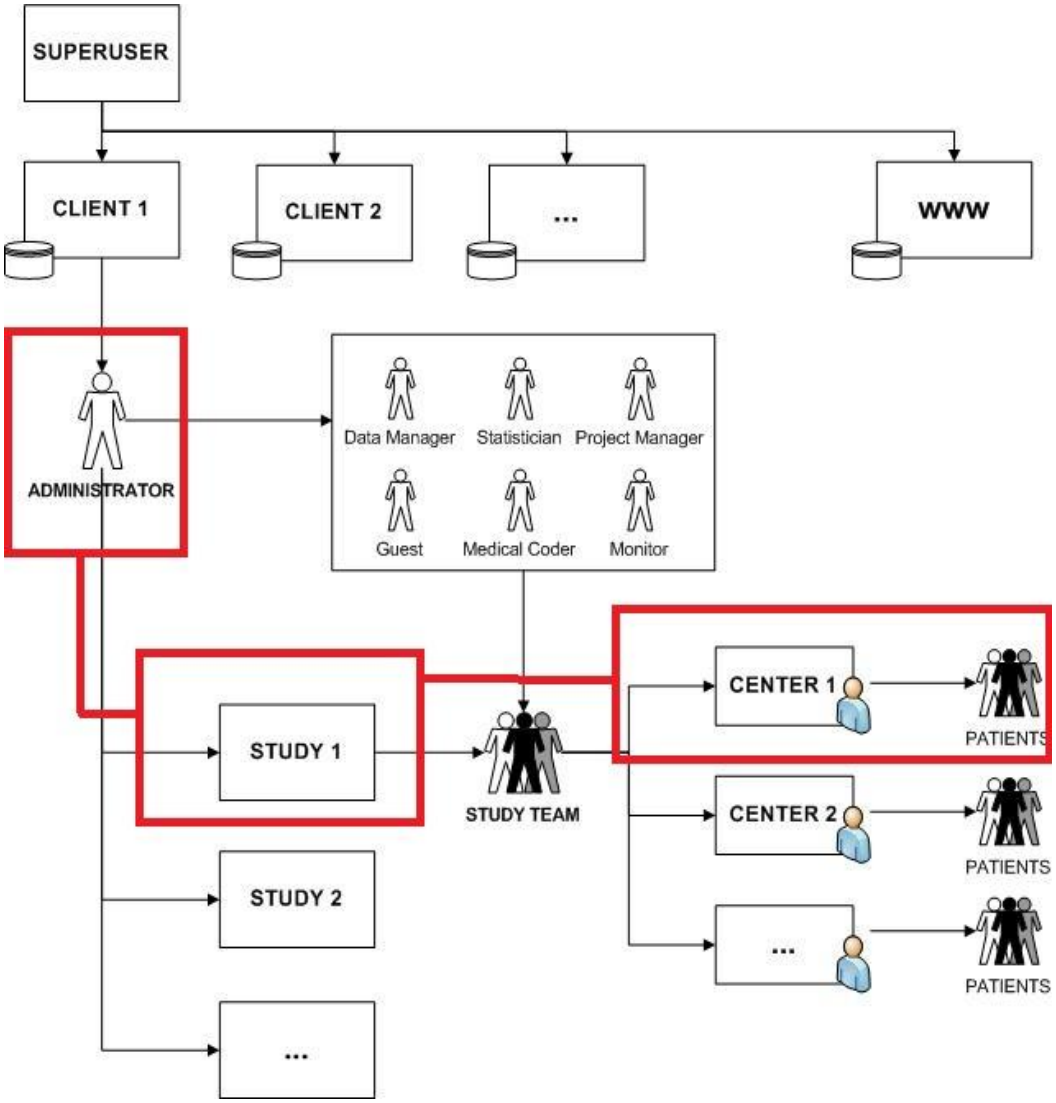


Ilustración 67- Conclusiones - Mejora multicentro.

No supondría un gran esfuerzo, traducir el programa a otros lenguajes. Hacerlo multilenguaje.

7. CONCLUSIONES

Este proyecto ha sido un proyecto ambicioso desde el principio, no se ha podido llegar a recoger datos del estudio, pero recoge todas las partes del proyecto desde la definición de la hoja de cálculo Excel Define.xls hasta la creación del estudio y la visualización en pantalla del Cuaderno de Recogida de Datos (CRF).
Se han invertido muchas horas de formación en el Framework CakePHP. En la comprensión y el funcionamiento. Pero hemos resuelto los imprevistos que han ido surgiendo.
En la parte conceptual, lo más complicación ha sido el diseño y las relaciones entre las diferentes tablas que componen la estructura de la hoja de EXCEL Define.xls. A continuación mostramos los posibles diseños que se han ido mejorando.

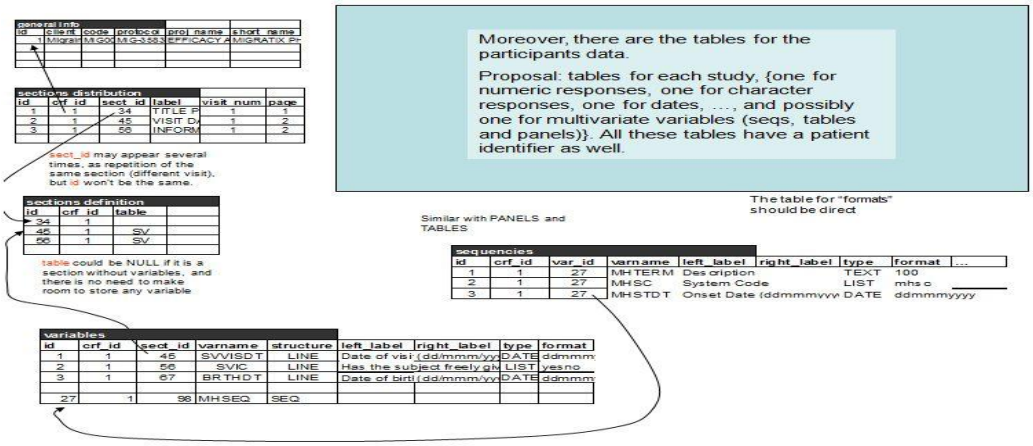


Ilustración 68 -Conclusiones - Mejoras del modelo conceptual I

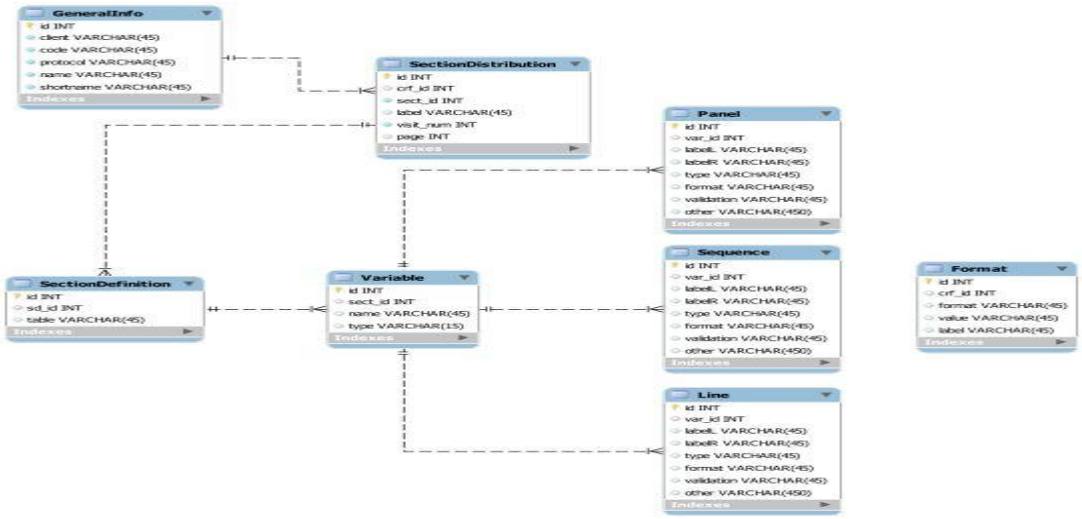


Ilustración 69-Conclusiones- Mejoras del modelo conceptual II

En cuanto a la parte gráfica, se ha querido hacer un diseño básico pero funcional. Se puede concluir, que los objetivos para los que se decidió desarrollar este proyecto se han alcanzado. Los conocimientos adquiridos en todos los campos del software son sólidos y se han podido poner en práctica en este proyecto real. Espero que sirva para la realización de otros proyectos.

Con el aprendizaje adquirido podríamos desenvolvernos, con más facilidad en un ámbito laboral.

Tras comprobar que la parte realizada de la aplicación funciona con total normalidad y eficacia, hay que admitir que todavía no está preparada para ser utilizada ya que no se recogen datos del estudio para llevar a cabo una explotación.

8. EXPERIENCIA Y VALORACIÓN PERSONAL

Como experiencia personal, lo primero que me ha aportado la realización de este proyecto ha sido el aprendizaje de diferentes lenguajes de programación como puede ser Jquery y herramienta como el Framework CakePHP.Tambien he podido asentar conocimiento de aquellos lenguajes y herramientas que ya conocía.

He podido comprobar la importancia de realizar una buena planificación del proyecto para poder ir corrigiendo los imprevistos que iban surgiendo.

Por último agradecer a todas las personas que me han ayudado y me han dado soporte durante la realización del proyecto como pueden ser la familia, el director, co- director del proyecto y el compañero de proyecto.

9. BIBLIOGRAFIA

[1] Manual PHP; *Framework* [en línea] [fecha de consulta: 01 de Diciembre de 2011] <http://es.php.net/manual/en/>

[2] CakePHP; *Framework* [en línea] [fecha de consulta: 02 de Diciembre de 2011] <http://www.cakephp.org/>

[3] CakePHP Cookbook v2.x; *Framework* [en línea] [fecha de consulta: 02 de Diciembre de 2011] <http://book.cakephp.org/2.0/en/contents.html>

[4] Synfony; *Framework* [en línea] [fecha de consulta: 02 de Diciembre de 2011] <http://www.symfony-project.com/>

[5] Php; *Framework* [en línea] [fecha de consulta: 03 de Diciembre de 2011] <http://www.phpframeworks.com/>

[6] Php; *Framework* [en línea] [fecha de consulta: 03 de Diciembre de 2011] <http://tuxpuc.pucp.edu.pe/articulo/comparativa-de-frameworks-en-php-cakephp-symfony-y-zend-framework>

[7] Canal Video -CakePHP; *Framework* [en línea] [fecha de consulta: 08 de Diciembre de 2011] <http://tv.cakephp.org/>

[8] Md5; [en línea] [fecha de consulta: 08 de Diciembre de 2011] < http://ansat.es/soporte/docs/md5/md5.htm>

[9] Autentificación CakePHP; [en línea] [fecha de consulta: 15 de Febrero de 2012] < http://aikon.com.ve/sistema-de-autenticacion-simple-en-cakephp/>

[10] Autentificación CakePHP; [en línea] [fecha de consulta: 16 de Febrero de 2012] < http://www.taringa.net/posts/videos/2521355/VideoTutorial-del-Curso-de-CakePHP-1-al-7.htm>

[11] PHP Excel; [en línea] [fecha de consulta: 17 de Febrero de 2012] < http://www.ideasfrescas.es/es/blog-paginas-web/web/articulos-web/excel-con-php.php>

[12] PHP Excel; [en línea] [fecha de consulta: 18 de Febrero de 2012] < http://phpexcel.codeplex.com/releases/view/88098>

[13] jQuery [en línea] [fecha de consulta: 17 de Abril de 2012]

< http://docs.jquery.com/Main_Pag>

[14] CakePHP: Authentication Setup on an Application [en línea] [fecha de consulta: 20 de Febrero de 2012] < http://docs.jquery.com/Main_Pag>

10. ANEXOS

ANNEX

ANNEX- 1 INTRODUCTION

DEFINE.XLS

SHEET - GENERAL INFO (GI)

SHEET - SECTIONS DISTRIBUTION (SDIS)

SHEET - SECTIONS DEFINITION (SDEF)

SHEET – FORMATS (FM)

ANNEX- 2 TOOLS USED

ANNEX- 3 CONCEPTUAL MODELS

ANNEX - 4 CLASS DIAGRAM

ANNEX- 5 USE CASE MODELS

ANNEX- 6 SEQUENCE DIAGRAM

ANNEX- 1 INTRODUCTION

A Case Report Form (CRF) is the document that the investigator uses to record all the information regarding a patient during a medical study or clinical trial. Common information to record in a clinical trial is demographic data, previous clinical history, concomitant medication and adverse events. Depending on the objective of the study, the CRF will incorporate specific medical assessments that may be repeated a long time, organized on visits, in order to assess if an intervention produces a change in the health of a patient.

In our_aplication, the CRF can be defined by means of either a wizard or an excel spreadsheet. This document explains how to generate CRFs using the excel spreadsheet, referred as Define.xls.

The information of a CRF is organized in *Define.xls* on different levels, by pages, sections, structures and variables, each of this nested in the former. Thus, a CRF is composed by one or more pages, each page contains one or more *sections*, each *section* contains one or more *structures* and each *structure* is composed by one of more *variables*.

As an example, the following Figure represents a common CRF page. This page is composed by 3 *sections*: Demographic Data, Smoking Habits and Alcohol Consumption. The *section* Demographic Data is composed by 5 different *structures*, where each *structure* in this case only contains one *variable*. Other *structures* can be defined in order to organize more complex data layouts like for instance a table with variables organized by rows and columns.

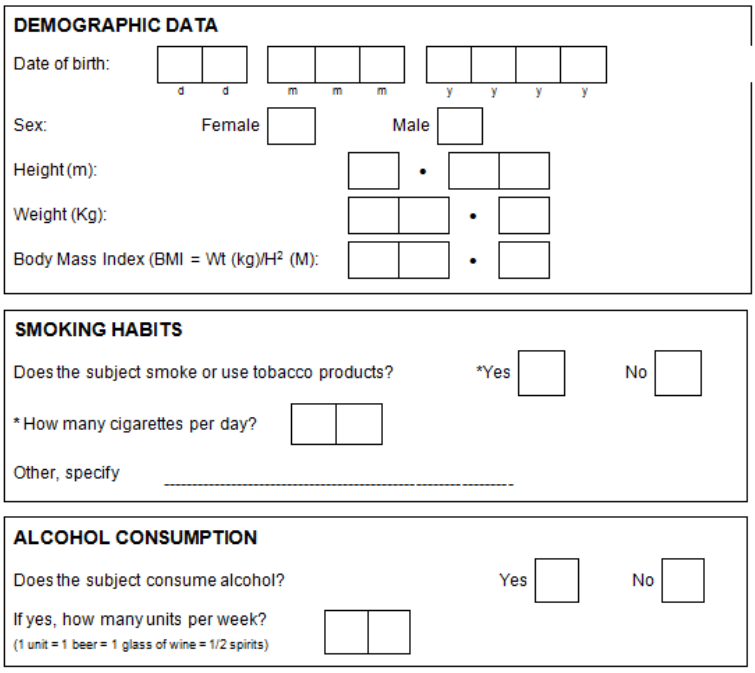


illustration 1- Common page of the CRF.

Define.xls is organized in 4 excel sheets: (1) Sections Distribution, (2) Sections Definition, (3) Formats, and (4) Message Texts. The first 3 sheets must be completed and the last is optional.

DEFINE.XLS

Sheet - General Info (GI)

In this spreadsheet defines the fields required for the study.

Parameter	Content
CLIENT	Migrain Pharmaceuticals
CODE	MIG001
PROTOCOL	MIG-35831
NAME	EFFICACY AND SAFETY PHASE III CLINICAL TRIAL OF ORAL MIGRATIX FORTHE TREATMENT OF SEVERE MIGRAINE
SHORTNAME	MIGRATIX PHASE III

illustration 2 -Example General Infos

Sheet - Sections distribution (SDIS)

In this spreadsheet is defined how the *sections* are distributed along the CRF on different pages. Each page can contain one or more *sections*. The *sections* can be repeated along the study using different page numbers. For example, it is pretended to measure the blood pressure of the patients on different visits; for that, a *section* called Blood Sample is defined and repeated on different pages (visits) using one row per each page [10].

Four columns must be defined:

Section	Label	Visit	Page
TPAGE	TITLE PAGE	1	1
VDATES	VISIT DATE	1	2
CONSENT	INFORMED CONSENT	1	2
DM	DEMOGRAPHIC DATA	1	2
SMOKE	SMOKING HABITS	1	2
ALC	ALCOHOL CONSUMPTION	1	2
MH	MEDICAL HISTORY	1	3
VS1	VITAL SIGNS	1	4
BS	BLOOD SAMPLE ASSESSMENT	1	4
VDATES	VISIT DATE	2	5
HIT	HIT-6	2	5
VAS	VAS	2	6
INC	INCLUSION CRITERIA	2	7
EXC	EXCLUSION CRITERIA	2	7
RAND	RANDOMIZATION	2	8
VS2	VITAL SIGNS	2	8
DRUG	STUDY DRUG INTAKE	2	8
VS3	VITAL SIGNS	2	8
VDATES	VISIT DATE	3	9
HIT	HIT-6	3	9
VAS	VAS	3	10
VDATES	VISIT DATE	4	11
HIT	HIT-6	4	11
VAS	VAS	4	12
VS1	VITAL SIGNS	4	12
EOS	END OF STUDY FORM	5	13
CM	CONCOMITANT MEDICATIONS	6	14
AE	ADVERSE EVENTS	6	15

illustration 3-Example Section distributions

Section: Abbreviated name to identify the *section*. These names will link to the spreadsheet *Sections Definition* where the contents of each *section* will be defined.

Label: Title of the section which will be showed as a header in the web form. If this field is omitted no header will be printed.

Visit: Each section must be related with a visit. The visit can be numeric or text. This field is mandatory.

Page: Indicates on which pages will be presented each section. If one section is repeated on different pages, a new row with the same section name and different page number must be added. This field is mandatory.

[10] Sections distribution (SDIS) ("Define v1.1.xls")[Figure 1]

Sheet - Sections definition (SDEF)

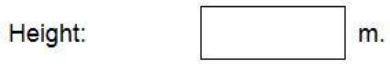
In this spreadsheet is defined the content of the *sections*. Each *section* must be defined only once. As introduced before, a *section* is formed by structures and variables. Since the variable is the smelled information unit that can be defined, each row on the excel will define one single variable. Then, by means of different properties given in the excel columns, the variables will be gathered constituting *sections* and *structures*[11].
The following columns are available:

Section: Name of *section,* previously assigned to a CRF page and visit in Sections Distribution. Must contain the same *section* name for all the variables that form the *section*. This field is mandatory.

Structure: This field is mandatory. Four different structures are defined: LINE, PANEL, SEQ and TABLE.

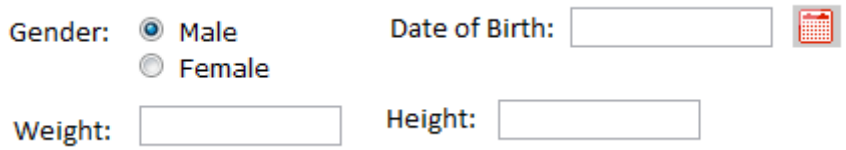
- **LINE**

The LINE *structure* is the simplest type of structure. It only contains one variable which is displayed on the WEB as simple line with the specified variable properties in other columns, for example,



- **PANEL**

The PANEL *structure* serves to organize multiple variables on a panel or matrix layout. In the following example, 4 variables constitute the PANEL organized in two rows and two columns.



The fist variable defined as PANEL must define the number of rows and columns of the PANEL. For that, it must be used the parameter *panel_dimensions* in the column *Other_parameters.* The variables will appear by row from left to right.

- **TABLE**

The TABLE *structure* gathers information using a table layout. The first variable defined as TABLE will correspond to the row identifier. This must have a format defined by the user where each value will correspond to one row. The order of the rows will be given by the format value, although the labels will be printed in each row.

[11] Sections definition (SDEF) (“Define v1.1.xls")[Figure 2]

The next variables will define the columns of the table. To identify unequivocally this data in the database, a new primary key will be added corresponding to the variable defined for the rows.
The headers of each column in the table are given in the column *left_label*.
To generate a table like below for Vital Signs, the first variable with TABLE *structure* must be Post_dose. This variable is associated with a user defined format which contains the levels: 1="1 hour", 2="2 hours" and 3="3 hours". See column Format for more information about user defined formats. The following variables defined as TABLE will be Date, Time, Systolic and Diastolic Blood Pressure.

Vital Signs

Post-dose	Date* [dd mmm yy]	Time [hh mm]	Blood Pressure Systolic / Diastolic [mmHg / mmHg]
1 hour			/
2 hours			/
3 hours			/

- **SEQ**

This structure organizes the variables in a similar way to the TABLE. In this case, however, the first variable will contain an automatically generated sequence (row number) instead of the values from a user defined format. In this way, there is no limitation on the number of rows. The following variables in the SEQ *structure* will define the columns of the table.

CONCOMITANT MEDICATIONS

Num.	Medication	Start Date (MM/DD/YYYY)	Stop Date (MM/DD/YYYY)
1.			
Add			

- **TEXT, TITLE1, TITLE2**

These structures do not affect the behavior of the variables, are just used for adding texts inside sections.
The text format can be changed using the column other_parameters, see: text_font, text_size, text_bold, text_italic, text_underline, text_just, text_color.

TITLE1 by default uses text_size = 5 and text_bold = TRUE.
TITLE2 by default uses text_size = 4 and text_bold = TRUE, text_italic=TRUE.

Example: section END OF STUDY contains four LINE structures with checkbox variables. Besides, different titles are added to group appropately the variables and a note at the end of the section is added.

END OF STUDY
End of Study Reason: **(TITLE1)** **(click only one reason)** ***(TITLE2)*** ☐ Completed study ☐ AE/SAE (complete AE CRF & SAE form, if applicable) **Other Reasons:** ***(TITLE1)*** ***(click only one reason)*** ***(TITLE2)*** ☐ Non-compliant participant ☐ Concomitant medication _ **Once the END OF STUDY page is completed, please freeze the patient if this is ready for review.** ***(TEXT structure with text_bold = TRUE, text_color = "blue")***

Variable: name of the variable; it is recommended to use the first two letters of the table which the variable belongs and not exceed 8 characters long. This field is mandatory.

The following columns are used to complete the information of each variable:

Table: this column indicates the name of the table in the database that will store this variable. It is recommended to use 2 letters to define the name of the table that will be always included in the name of the variable. This field is mandatory.

Left_label: text of the question that will be showed in the web form on the left side of the field to introduce the answer. See the column *other_parameters* for further options on how to modify the appearance of this label. This field is optional.

Right_label: text to show on the right side of the answer field; commonly used indicate the unit or format of the answer, e.g, meters or date format as dd-mm-yy. This field is optional.

Type and Format: The columns type and format work together to specify how to retrieve and present the information.

Type	Format	Description	Example
NUM	*n* *n.d*	Numeric variables where *n* represent number of digits for the integer part and *d* the number of decimals.	Height 1.65 m.
TEXT	*n*	Text variables, n indicates the number of character that are allowed.	Sex: Female
DATE	*ddmmmyyyy* *ddmmyyyy* *ddmmmyy* *ddmmyy*	Date format. Two options for the month: *mm* for numeric format (01-12), or *mmm* for text format (JAN-DEC). Two options for the year presentation: yyyy for the full year, e.g. 2012, and yy for only the last two digits, e.g., 12.	ddmmmyyyy 01-JAN-2012 ddmmyy 01-01-12
PDATE	*Idem DATE*	Like date but in this case partial dates are allowed, e.g, if ddmmyy is specified in format column, users can also provide month-year or only year. See Date for more details on data formats.	ddmmmyyyy 01-JAN-2012 JAN-2012 2012
TIME	*hh: mm*	Time 24 hours format.	
LIST	*User_format*	Drop down list, allows one of multiple choices given by a user defined format, see spreadsheet Formats.	Gender: Female Male
RBUTTON	*User_format*	Radio button, allows one of multiple choices given by a user defined format, see spreadsheet Formats.	Gender: Male Female
CHECKBOX	*User_format*	Checkbox, allows one or multiple choices among the values given by a user defined format, see spreadsheet Formats.	Treatment A Treatment B
DERIVED	*n* *n.d* *date??*	This type of input indicates that the content will be automatically calculated given information from other fields. The format must specify how to present the result, see numeric format.	

Validation:

If an answer is given, one of the following functions can be specified to perform a validation:

- ***in (value1, value2, …):*** the value must be contained among the given list of values, e.g. score bust be 1, 2, 3 or 99, in(1,2,3,99) .
- ***gt(value):*** greater than a value, e.g. must be heavier than 40 kg, gt(40).
- ***ge(value):***greater equal to a value, e.g. must be at least 18 years old, gt(18).
- ***lt(value):*** less than a value, e.g. must be younger than 18 years old, lt(18).
- ***le(value):*** less or equal to a value,e.g. must be smaller or equal to 220 cm, le(220).
- ***bw(min,max):*** between min and max values, both included; e.g. age must be between 0 and 120, bw(0,120).

Comments:

By default, all fields defined as dates are automatically validated. To avoid this, *validate_date = FALSE* must be specified in the column *Other_parameters*.

Other_parameters:

Several parameters may be together, separated by semicolons ";"

- ***Formula:*** formula=*expression*, being expression a mathematical formula comprehensible on Javascript for calculating the value for a Derived type variable.
- ***left_input:*** if TRUE the input field is presented before the label; the default is FALSE.
- ***open_if:*** *logical condition,* if true, the field will be open and will be possible to introduce information; if false, the field will be locked.
- ***show_if:*** *logical condition,* if true, the question will appear; hidden otherwise.
- ***close_if:*** *logical condition,* if true, the field will be locked; open otherwise.
- ***col_size:*** percentage of table
- ***input_size:*** specify the width of an input in number of characters.
- ***required:*** if TRUE, the field must be answered before saving the data of that *section*.

Text options: the following options are applied to the text specified left_label (SDEF).

- ***text_font:*** one of the following: verdana, arial, courier,html available fonts?
- ***text_bold*:** TRUE for bold text, otherwise FALSE (default).
- ***text_italic:*** TRUE for italic text, otherwise FALSE (default).
- ***text_underline:*** true for undelined text, otherwise FALSE (default)
- ***text_size:*** text size, from 1 (smallest) to 7 (biggest).

- ***text_just*:** justification for TEXT structures. One of the followings: left (default), right, center.

text_color: color for TEXT structures. One among the following: black (default), green, red, blue, yellow, grey, ???

validate_date: FALSE to omit the automatic validation of the dates.

Error and Help messages:

Variables with HELP messages will be shown with a "?" sign aside, so that the message appears when the user moves the mouse over it.

Variables with ERROR messages need to be linked to a validation expression. When it does not hold, the error message raises up. These messages can override default messages, if implemented. For instance: validation expression in() could have a default message stating something like "The input must be one of … (the list between parentheses)".

Section	Structur	Variable	Table	Left_label	Right_label	Type	Format	Validation	Other_Parameters	message_error	message_help
VDATES	LINE	SVVISDT	SV	Date of visit	(dd/mmm/yyyy)	DATE	ddmmmyyyy		required=TRUE		
CONSENT	LINE	SVIC	SV	Has the subject freely given written informed consent?		LIST	yesno	in(1)			
DM	LINE	BRTHDT	DM	Date of birth	(dd/mmm/yyyy)	DATE	ddmmmyyyy				
DM	LINE	SEX	DM	Sex		LIST	sex				
DM	LINE	HEIGHT	DM	Height	m	NUM	1.2	bw(1,2.5)			
DM	LINE	WEIGHT	DM	Weight	Kg	NUM	3.1	bw(20,300)			
DM	LINE	BMI	DM	BMI	Kg/m2	DERIVED	2.2		formula=round(WEIGHT/HEIGHT^2,2)		
SMOKE	LINE	DMSMOKE	DM	Does the subject smoke or use tobacco products?		LIST	yesno				
SMOKE	LINE	DMSMOKE	DM	How many cigarettes per day?		NUM	2	bw(0,199)	open_if = (DMSMOKE=1)		
SMOKE	LINE	DMSMOKE0	DM	Other, specify		TEXT	200		open_if = (DMSMOKE=1); input_size=50:2		
ALC	LINE	DMALC	DM	Does the subject consume alcohol?		LIST	yesno				
ALC	LINE	DMALCQ	DM	If yes, how many units per week?		NUM	2.1		open_if = (DMALC=1)		
MH	LINE	MHPRES	MH	Does the subject present with, or has a past relevant history of any condition faillling into any of the following categories?		LIST	yesno				
MH	SEQ	MHSEQ	MH								
MH	SEQ	MHTERM	MH	Description		TEXT	100				
MH	SEQ	MHSC	MH	System Code		LIST	mhsc				
MH	SEQ	MHSTDT	MH	Onset Date	(ddmmmyyyy)	DATE	ddmmmyyyy				
MH	SEQ	MHENDT	MH	End Date	(ddmmmyyyy)	DATE	ddmmmyyyy				
MH	SEQ	MHPREVI	MH	Prevent inclusion		LIST	yesno				
VS1	LINE	VSHR	VS	Heart Rate	bpm	NUM	3	bw(30,180)			
VS1	LINE	VSSBPSC	VS	Systolic blood pressure	mmHg	NUM	2.1				
VS1	LINE	VSDBPSC	VS	Diastolic blood pressure	mmHg	NUM	2.1				
BS	LINE	BSBS	BS	Has the blood sample been collected?		RBUTTON	yesno				
BS	LINE	BSBSDT	BS	Please specify date:		DATE	ddmmmyyyy		show_if = (BSBS=1)		
BS	LINE	BSBSOTH	BS	Please specify a reason:		TEXT	100		show_if = (BSBS=0)		
HIT	LINE	QLHIT1	QL	1. When you have headaches, how often is the pain severe?		LIST	hit				
HIT	LINE	QLHIT2	QL	2. How often do headaches limit your ability to do usual daily activities including household work, work, school, or social activities?		LIST	hit				
HIT	LINE	QLHIT3	QL	3. When you have a headache, how often do you wish you could lie down?		LIST	hit				
HIT	LINE	QLHIT4	QL	4. In the past 4 weeks, how often have you felt too tired to do work or daily activities because of your headaches?		LIST	hit				
HIT	LINE	QLHIT5	QL	5. In the past 4 weeks, how often have you felt fed up or irritated because of your headaches?		LIST	hit				
HIT	LINE	QLHIT6	QL	6. In the past 4 weeks, how often did headaches		LIST	hit				

Sheet – Formats (FM)

In case any variable with type LIST, RADIO_BUTTON or CHECKBOX in included in the CRF, it is necessary to include in this spreadsheet the associated formats[12].
Three columns must be completed:

- ***Format:*** name for the format.
- ***Values:*** numeric value to order the possible levels of the format.
- ***Labels:*** levels that will be showed as the different option in the LIST, RADIO_BUTTON or CHECKBOX.

Format	Values	Labels
sex	1	Male
sex	2	Female
race	1	Caucasian
race	2	Other
yesno	1	Yes
yesno	0	No
tick	1	Yes
tick	2	No
mhsc	1	Neurological
mhsc	2	Psychiatric
mhsc	3	Eyes/Ears/Nose/Throat
mhsc	4	Skin
mhsc	5	Cardiovascular
mhsc	6	Respiratory
mhsc	7	Hepato-Gastrointestinal
mhsc	8	Musculoskeletal
mhsc	9	Genito-Urinary
mhsc	10	Allergic (other rhinitis)
mhsc	11	Endocrine/Metabolic
mhsc	12	Hematological/Lymphatic
mhsc	13	Immunological
mhsc	14	Other
cmroute	1	Oral
cmroute	2	Intravenous
cmroute	3	Intramuscular
cmroute	4	Sub-cutaneous
cmroute	5	Intradermal
cmroute	6	Rectal
cmroute	7	Inhalatory
cmroute	8	Ocular
cmroute	9	Transdermal
cmroute	10	Topical
cmroute	11	Nasal
cmroute	12	Otic
cmroute	13	Other

[12] Formats (FM) (“Define v1.1.xls")[Figure 3]

aesev	1	Mild
aesev	2	Moderate
aesev	3	Sever
aeser	1	Serious
aeser	2	Not serious
aerel	1	Not related
aerel	2	Unlikely
aerel	3	Possible
aerel	4	Probable
aerel	5	Certain
aerel	6	Unassessable
aeout	1	Completely recovered
aeout	2	Recovered with sequelae
aeout	3	Persistence
aeout	4	Death
aeout	5	Unknown
aeacn	1	None
aeacn	2	Decrease of dose
hit	1	Never
hit	2	Rarely
hit	3	Sometimes
hit	4	Very Often
hit	5	Always
vas	0	0 - Not at all
vas	1	1 - Mildly
vas	2	2 - Mildly
vas	3	3 - Mildly
vas	4	4 - Moderately
vas	5	5 - Moderately
vas	6	6 - Moderately
vas	7	7 - Marketdly
vas	8	8 - Marketdly
vas	9	9 - Marketdly
vas	10	10 - Extremly
tick_y	1	Not worked/studied during the last week
bptimes	1	1 hour
bptimes	2	2 hours
bptimes	3	3 hours
bptimes	4	4 hours
bptimes	5	5 hours
bptimes	6	6 hours
bptimes	8	8 hours
bptimes	10	10 hours
bptimes	12	12 hours
Visit_num	1	VISIT 1 (SCREENING)
Visit_num	2	VISIT 2 (RANDOMIZATION)
Visit_num	3	VISIT 3 (MONTH 3)
Visit_num	4	VISIT 4 (MONTH 6)
Visit_num	5	END OF STUDY FORM
Visit_num	6	AEs & CMs

illustration 4- Example Formats

ANNEX - 2 TOOLS USED

XAMPP

XAMPP is a package of Apache web server, a MySQL database and PHP interpreters and Perl languages. In fact the name comes from there, X (for any operating system), A (Apache), M (MySQL), P (PHP) and P (Perl). XAMPP is platform independent and is licensed under GNU GPL. There are versions for Linux (tested for SuSE, RedHat, Mandrake and Debian), Windows (Windows 98, NT, 2000, XP and Vista), MacOS X and Solaris (developed and tested with Solaris 8, tested with Solaris 9).
One advantage of XAMPP is a very simple and fast (no more than 5 minutes) you can ride your machine in a development environment of any Web application that uses PHP and database. The default configuration of XAMPP has some security weaknesses so it is not recommended for use as a production tool, but with some modifications is safe enough for use as a server on internet websites. From LAMPP (Linux AMPP) if you can make a secure facility by "/ opt / lampp / lampp security".

INSTALLATION

Obviously the first thing we have to go to the official website and download the XAMPP installer. For this tutorial we will install XAMPP on a Windows XP Pro and we will use the installer (http://www.apachefriends.org). Also be installed without unpacking the ZIP installer directly on your machine.

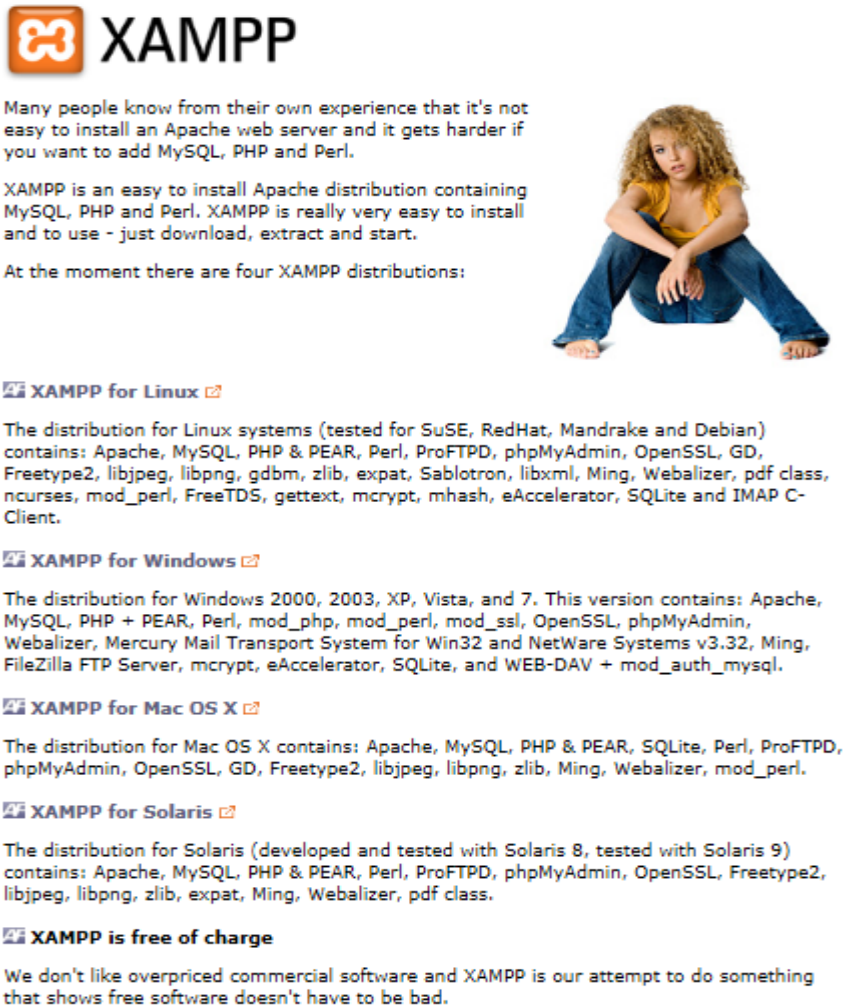
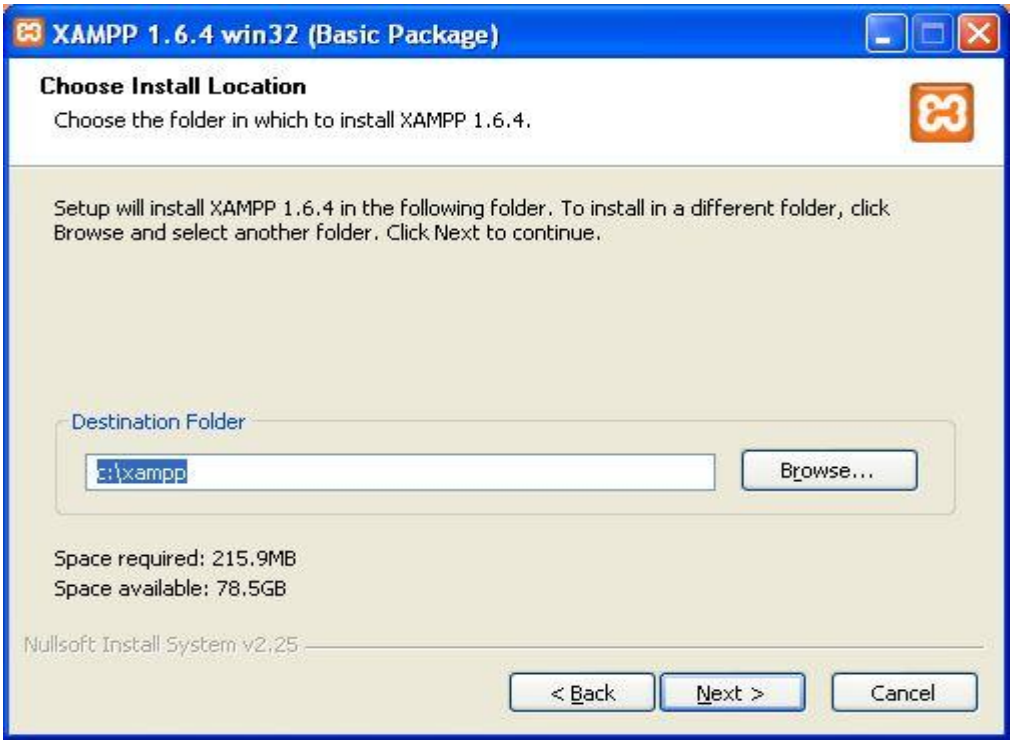


illustration 5- Xampp- http://www.apachefriends.org/es/xampp.html

After downloading the installer we will begin to install.

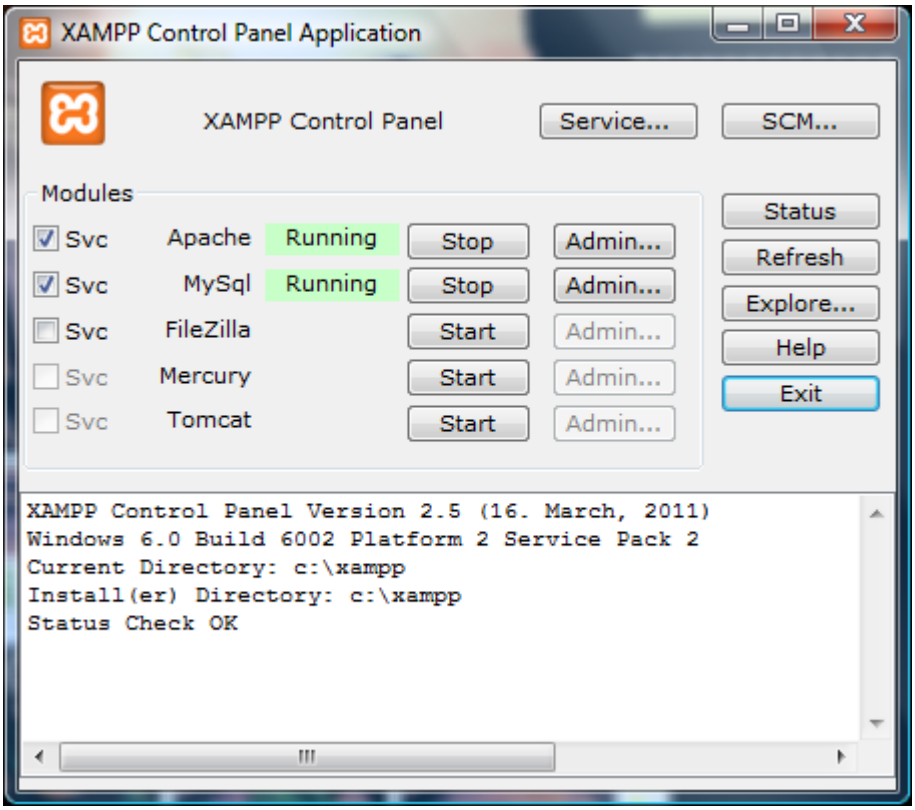
Next.

Next.



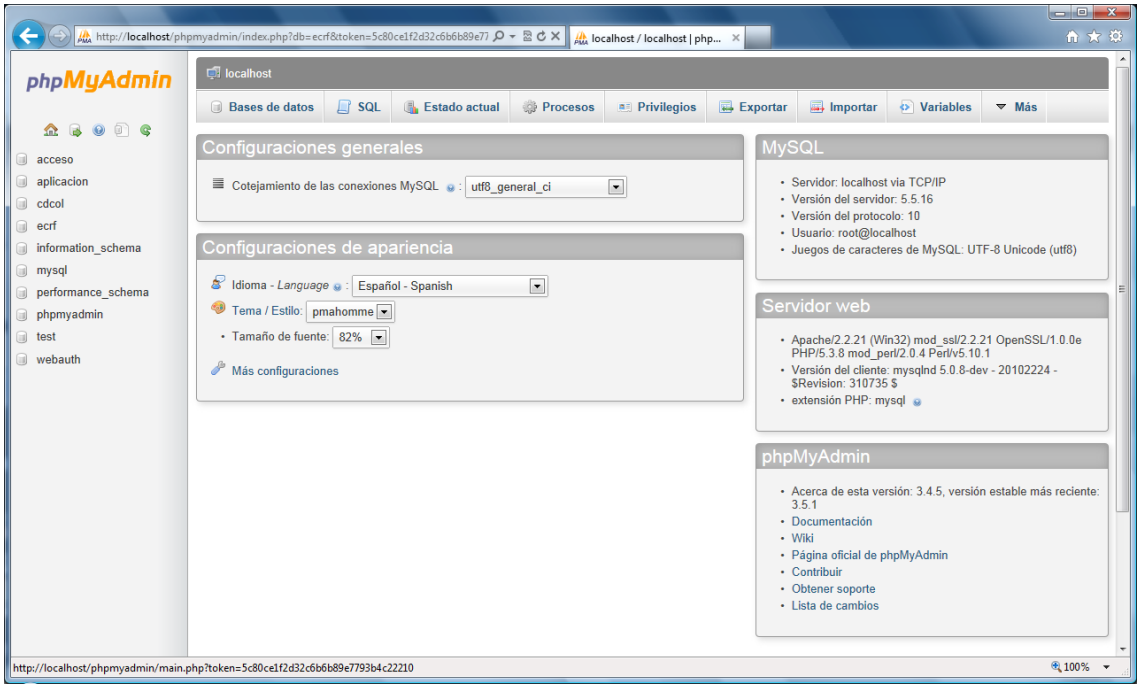
ADMINISTRATOR

In this control panel you can see all installed modules. For each module can stop your service (Stop), start it (Start), view the status (Stop / Running), mark it as a service (checkbox Svc) and enter your admin panel (Admin).



To prove the XAMPP installation was successful enough to set your browser to "http://localhost" or "http://127.0.0.1" and we will see the web administration application. Here we have a web administration section of XAMPP.

Create the database phpMyAdmin.



Now enter the cake folder *C: \Program Files\ xampp\ htdocs\xampp\cake\app\config* and here we *database.php.default* copy of the file and rename it in *database.php*, open it and do the connectivity to the database, the class will appear as follows *database_config*.

```
class DATABASE_CONFIG {

	public $default = array(
		'datasource' => 'Database/Mysql',
		'persistent' => false,
		'host' => 'localhost',
		'login' => 'root',
		'password' => '',
		'database' => 'ecrf',
		'prefix' => '',
		//'encoding' => 'utf8',
	);

	public $ecrf = array(
		'datasource' => 'Database/Mysql',
		'persistent' => false,
		'host' => 'localhost',
		'login' => 'root',
		'password' => '',
		'database' => 'ecrf',
		'prefix' => '',
		//'encoding' => 'utf8',
	);
}
```

ANNEX - 3 CONCEPTUAL MODELS

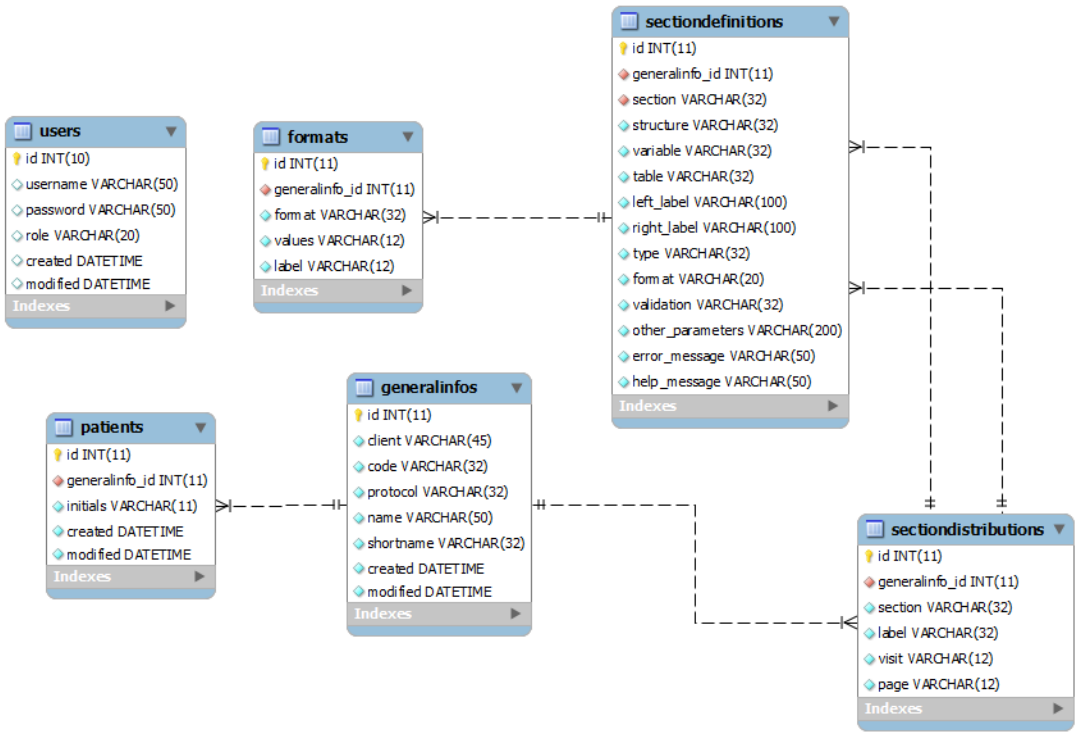


illustration 6 – Conceptual Model

TABLES

Table generalinfos

```
CREATE TABLE IF NOT EXISTS `generalinfos` (
`id` int(11) NOT NULL AUTO_INCREMENT,
`client` varchar(45) NOT NULL,
`code` varchar(32) NOT NULL,
`protocol` varchar(32) NOT NULL,
`name` varchar(50) NOT NULL,
`shortname` varchar(32) NOT NULL,
`created` datetime NOT NULL,
`modified` datetime NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=1;
```

Table sectiondistributions

```
CREATE TABLE IF NOT EXISTS `sectiondistributions` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `generalinfo_id` int(11) NOT NULL,
  `section` varchar(32) NOT NULL,
  `label` varchar(32) NOT NULL,
  `visit` varchar(12) NOT NULL,
  `page` varchar(12) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `generalinfo_id` (`generalinfo_id`),
  KEY `section` (`section`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
```

Table sectiondefinitions

```
CREATE TABLE IF NOT EXISTS `sectiondefinitions` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `generalinfo_id` int(11) NOT NULL,
  `section` varchar(32) NOT NULL,
  `structure` varchar(32) NOT NULL,
  `variable` varchar(32) NOT NULL,
  `table` varchar(32) NOT NULL,
  `left_label` varchar(100) NOT NULL,
  `right_label` varchar(100) NOT NULL,
  `type` varchar(32) NOT NULL,
  `format` varchar(20) NOT NULL,
  `validation` varchar(32) NOT NULL,
  `other_parameters` varchar(200) NOT NULL,
  `error_message` varchar(50) NOT NULL,
  `help_message` varchar(50) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `section` (`section`),
  KEY `generalinfo_id` (`generalinfo_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
```

Table patients

```
CREATE TABLE IF NOT EXISTS `patients` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `generalinfo_id` int(11) NOT NULL,
  `initials` varchar(11) NOT NULL,
  `created` datetime NOT NULL,
  `modified` datetime NOT NULL,
  PRIMARY KEY (`id`),
  KEY `generalinfo_id` (`generalinfo_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
```

Table formats

```
CREATE TABLE IF NOT EXISTS `formats` (
 `id` int(11) NOT NULL AUTO_INCREMENT,
 `generalinfo_id` int(11) NOT NULL,
 `format` varchar(32) NOT NULL,
 `values` varchar(12) NOT NULL,
 `label` varchar(12) NOT NULL,
 PRIMARY KEY (`id`),
 KEY `generalinfo_id` (`generalinfo_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
```

Table users

```
CREATE TABLE IF NOT EXISTS `users` (
 `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
 `username` varchar(50) DEFAULT NULL,
 `password` varchar(50) DEFAULT NULL,
 `role` varchar(20) DEFAULT NULL,
 `created` datetime DEFAULT NULL,
 `modified` datetime DEFAULT NULL,
 PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=7 ;
```

ANNEX - 4 CLASS DIAGRAM

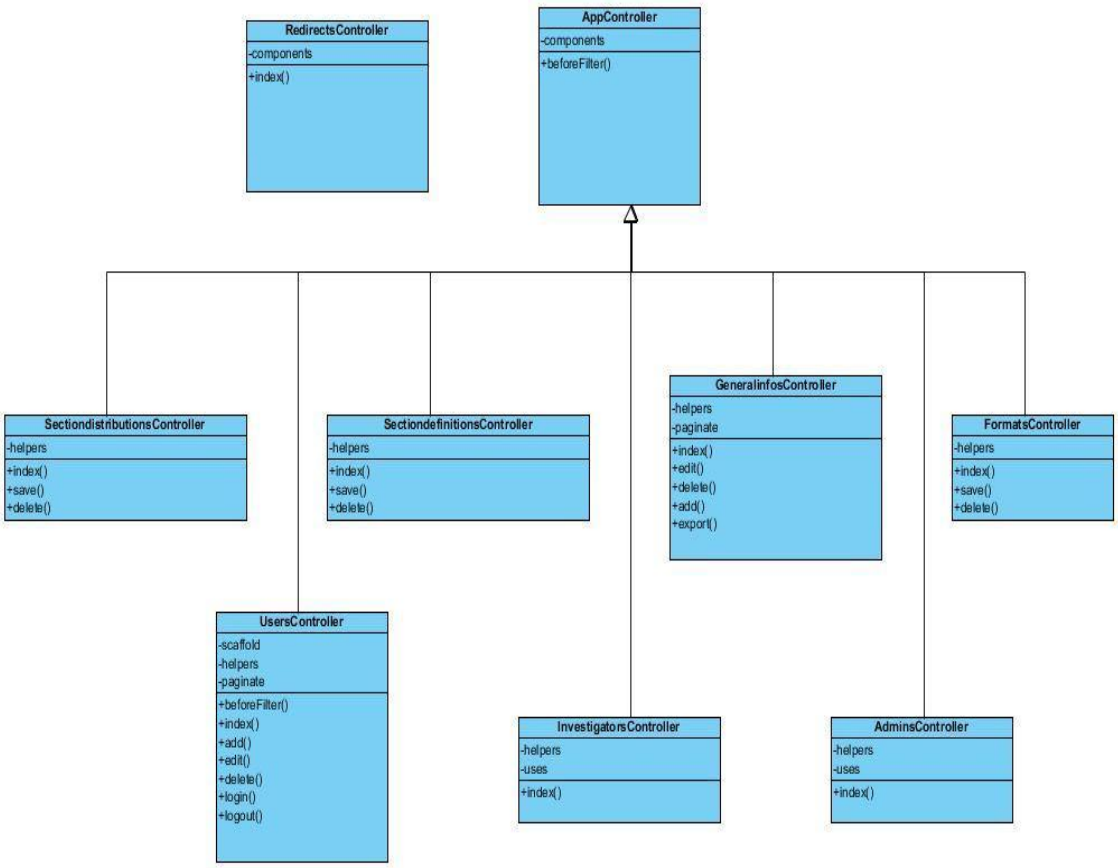


illustration 7 – Class Diagram

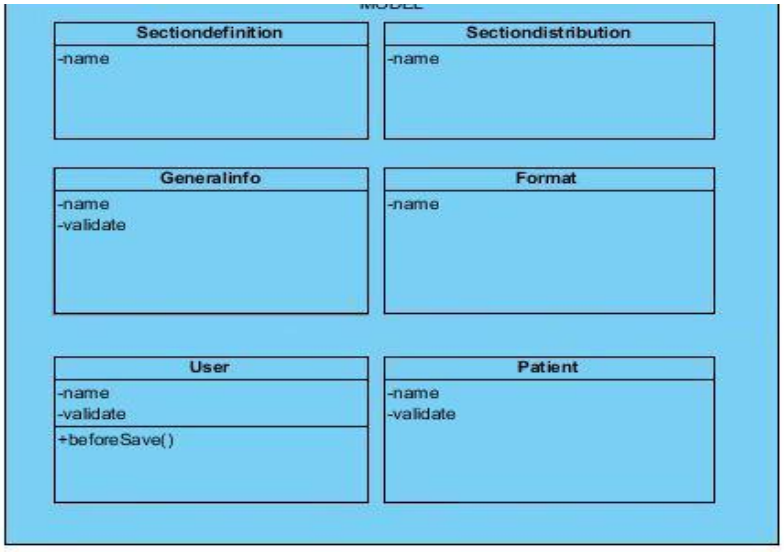


illustration 8- Class Diagram II

ANNEX - 5 USE CASE MODELS

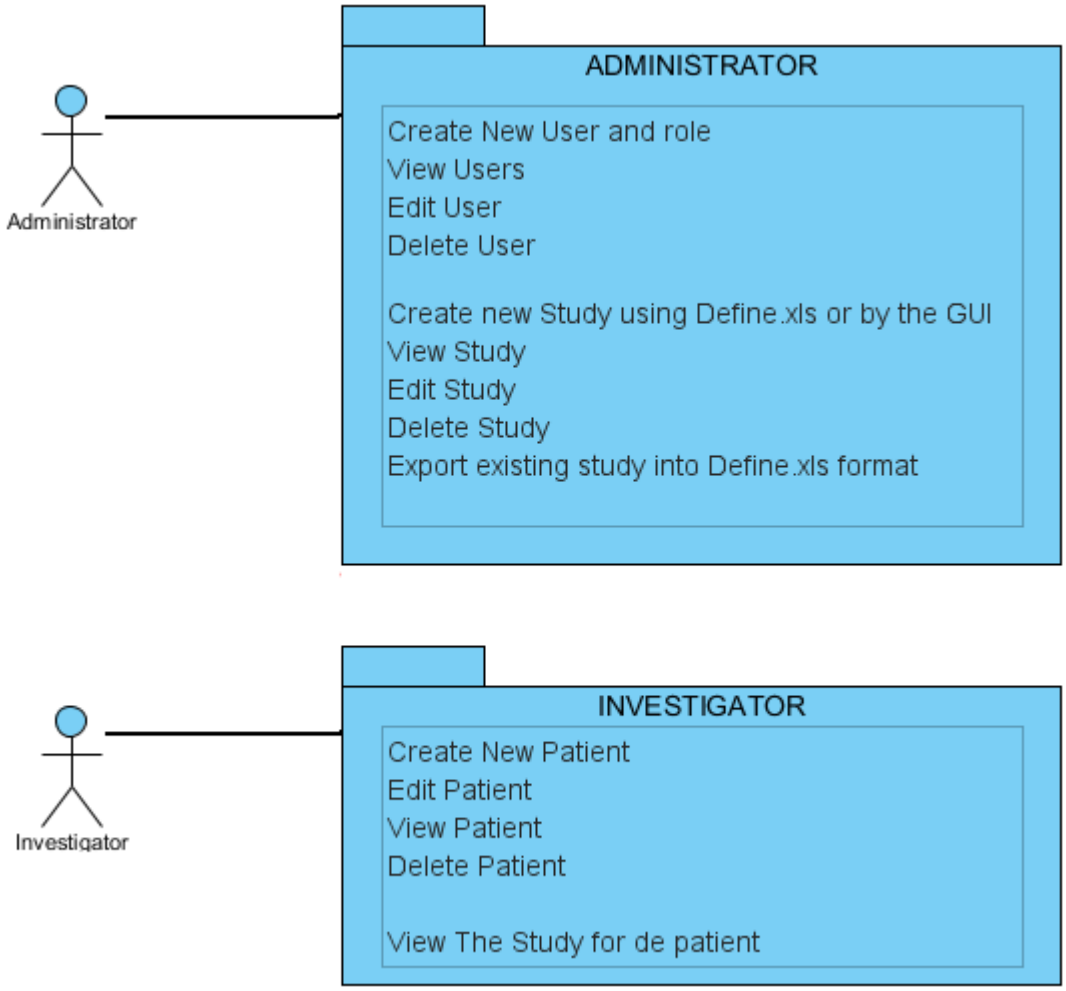


illustration 9 - Use Case Model

ADMINISTRATOR

Manage users:

- Create New User
- View Users
- Edit User
- Delete User

Use case: Create new user **Actor: Administrator**	
Action Manager	**System Response**
1. The administrator creates a new user data providing new user.	2. Verify that the user does not exist.
	3a.If there is generated error. 3b.If there is no save data for the new user.

Illustration 700 - USE CASE MODELS: Create User

Use case: List User Actor: Administrator	
Action Manager	**System Response**
1. The administrator requests a list of users.	2. Displays the users list.

Illustration 11 - USE CASE MODELS: List User

Use case: Edit User Actor: Administrator	
Action Manager	**System Response**
1. The administrator creates a new user data providing new user.	2. Verify that the user does not exist.
	3a.If there is generated error. 3b.If there is no save data for the new user.

Illustration 12 - USE CASE MODELS: Edit User

Use case: Delete user Actor: Administrator	
Action Manager	**System Response**
1. The administrator deletes the selected user.	2. Delete confirmation message.

Illustration 13 - USE CASE MODELS: Delete User

Manage studies:

- Create new Study using Define.xls
- View Study
- Edit Study
- Manage study versions
- Delete Study
- Export existing study into Define.xls format

Use case: Create Study Actor: Administrador	
Action Manager	**System Response**
1. The administrator creates a new study by selecting the EXCEL spreadsheet.	2. Create the study.

Illustration 14 - USE CASE MODELS: Create study

Use case: List study Actor: Administrator	
Action Manager	**System Response**
1. The manager asks to see the study.	2. Displays data from the study.

Illustration 15 - USE CASE MODELS: List study

Use case: Edit study Actor: Administrator	
Action Manager	**System Response**
1. The administrator selects the study to modify the overview of the study.	2. Check that there is no study with the same name.
	3a.If there is generated error. If there is no 3b.general stores data of the study.

Illustration 16 - USE CASE MODELS: Edit study

Use case: Delete Study Actor: Administrator	
Action Manager	**System Response**
1. The administrator deletes the selected study.	2. The system asks if the study is that you want to delete.
	3. Confirms that the study has been eliminated.

Illustration 17 - USE CASE MODELS: Delete Study

INVESTIGATOR

Manage patients:

- Create New Patient
- Edit Patient
- View Patient
- Delete Patient

Use case: Create patient Actor: Researcher	
Action Research	**System Response**
1. The researcher creates a new patient, providing new patient data.	2. Check that the patient does not exist.
	3a. If there is generated error. 3b. If there is no save data of the new patient.

Illustration 18 - USE CASE MODELS: Create patient.

Use case: List patients Actor: Researcher	
Action Research	**System Response**
1. The administrator requests a list of patients.	2. Displays the list patients.

Illustration 19 - USE CASE MODELS: List de patient

Use case: Edit patient Actor: Researcher	
Action Research	**System Response**
1. The administrator creates a new patient, providing the data.	2. Check that the patient does not exist.
	3a.If there is generated error. If there is no 3b.Save data of the new patient.

Illustration 20 - USE CASE MODELS: Edit patient

Use case: Delete patient Actor: Researcher	
Action Research	**System Response**
1. The administrator deletes the selected patient.	2. Confirmation message.

Illustration 21 - USE CASE MODELS: Delete patient

ANNEX - 6 SEQUENCE DIAGRAM

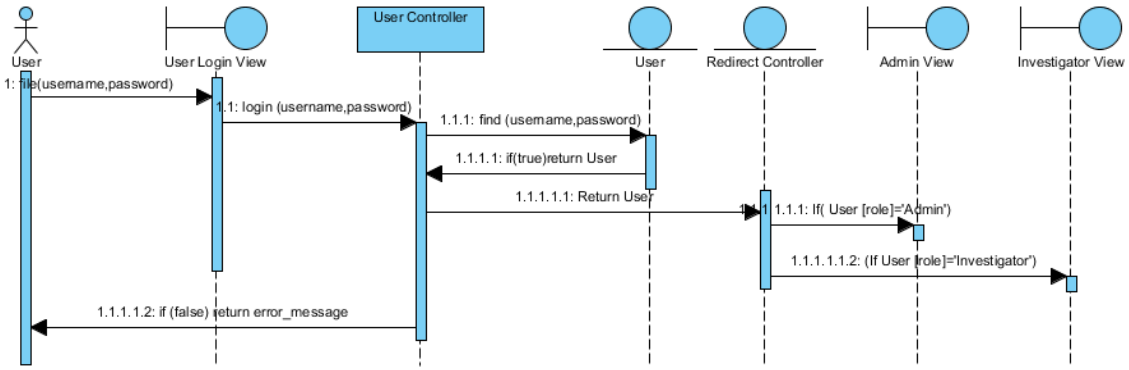


Illustration 22- SEQUENCE DIAGRAM -Login

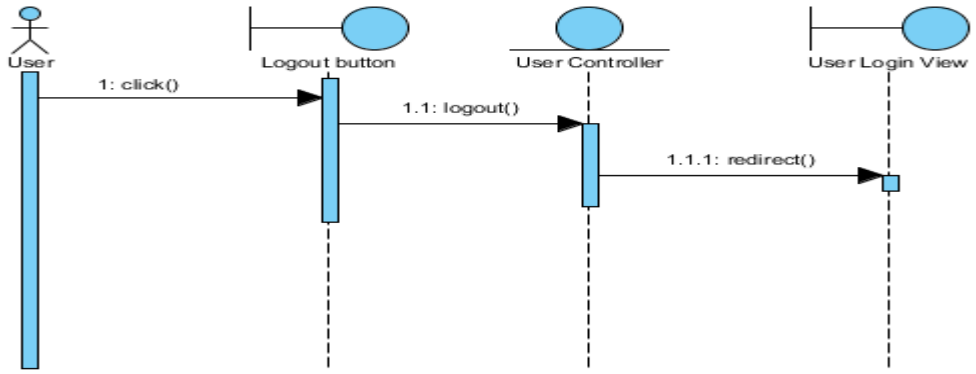


Illustration 23- SEQUENCE DIAGRAM - Logout

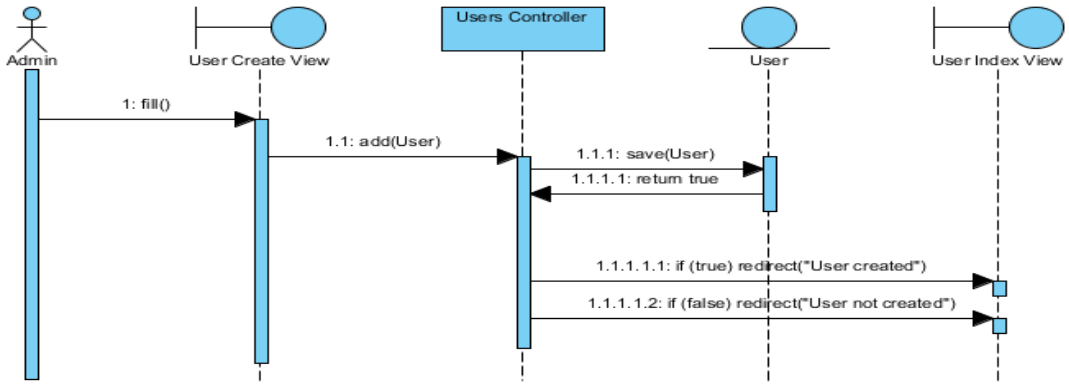


Illustration 24- SEQUENCE DIAGRAM -Create User

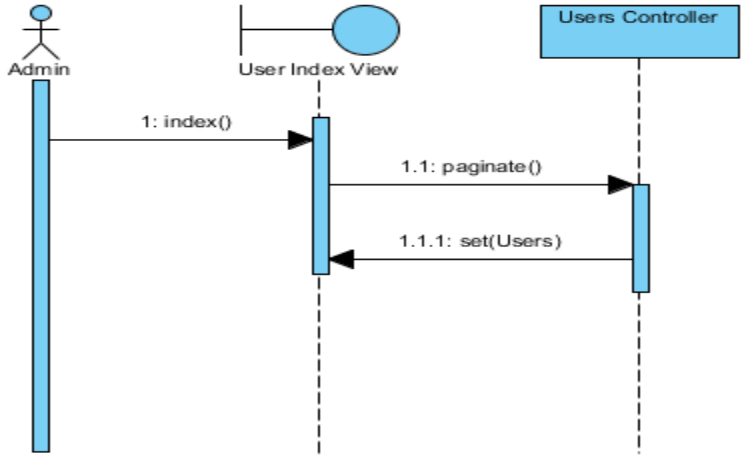


Illustration 25- SEQUENCE DIAGRAM -Index User

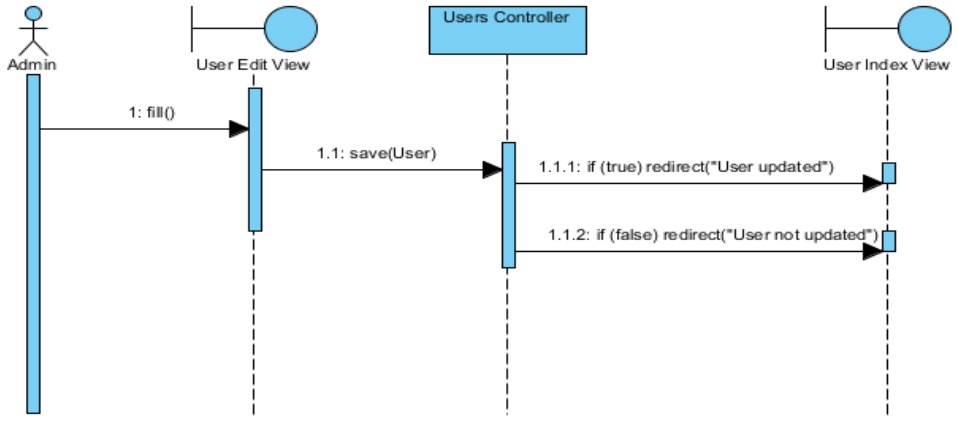


Illustration 26- SEQUENCE DIAGRAM- Edit User

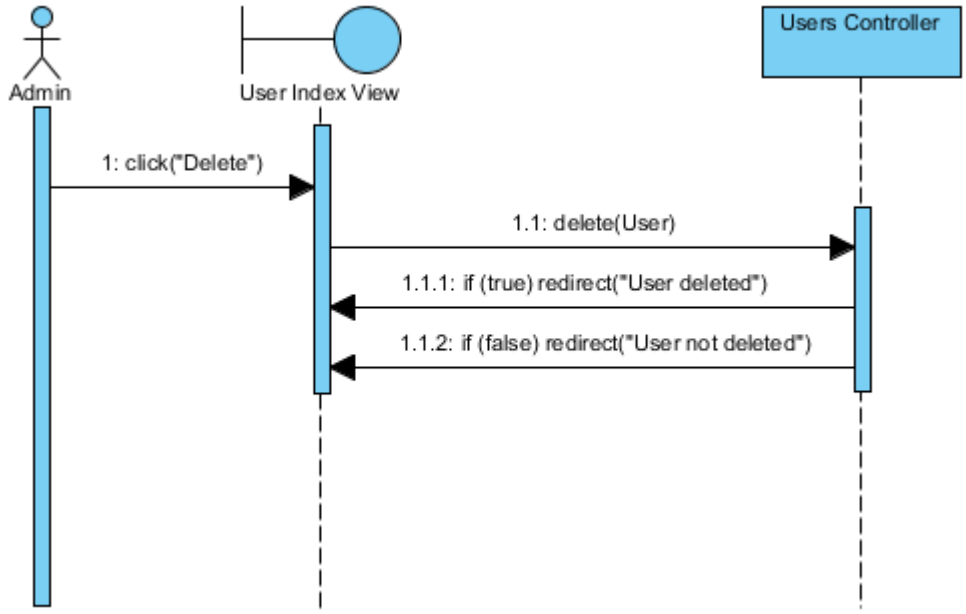


Illustration 27 - SEQUENCE DIAGRAM - Delete User

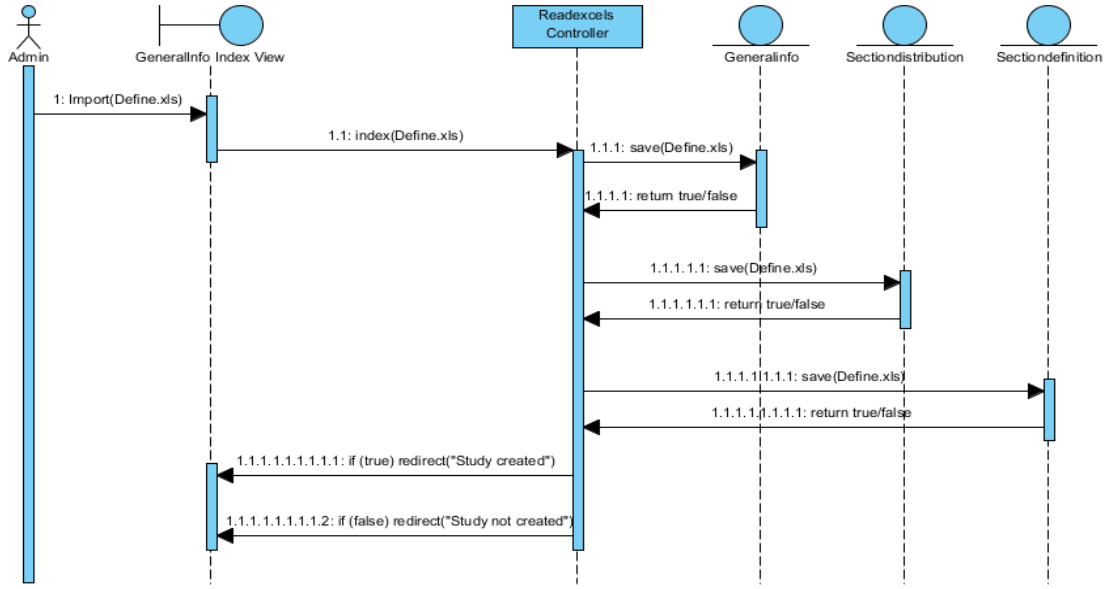


Illustration 28- SEQUENCE DIAGRAM -Create study

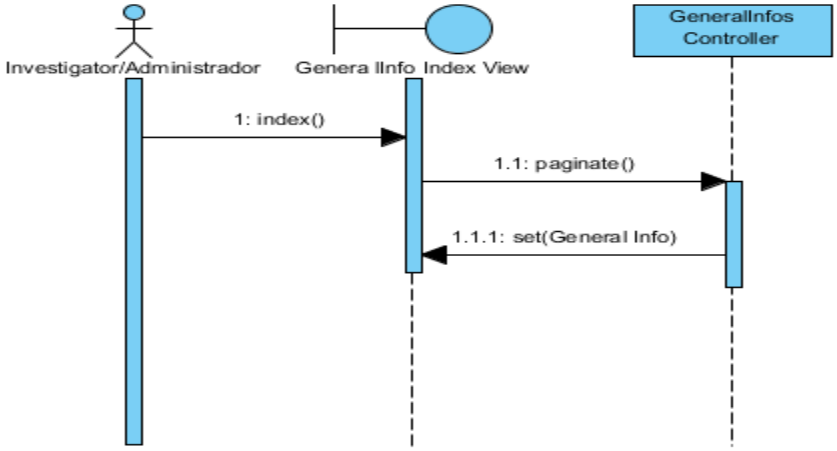


Illustration 29 - SEQUENCE DIAGRAM - View Study

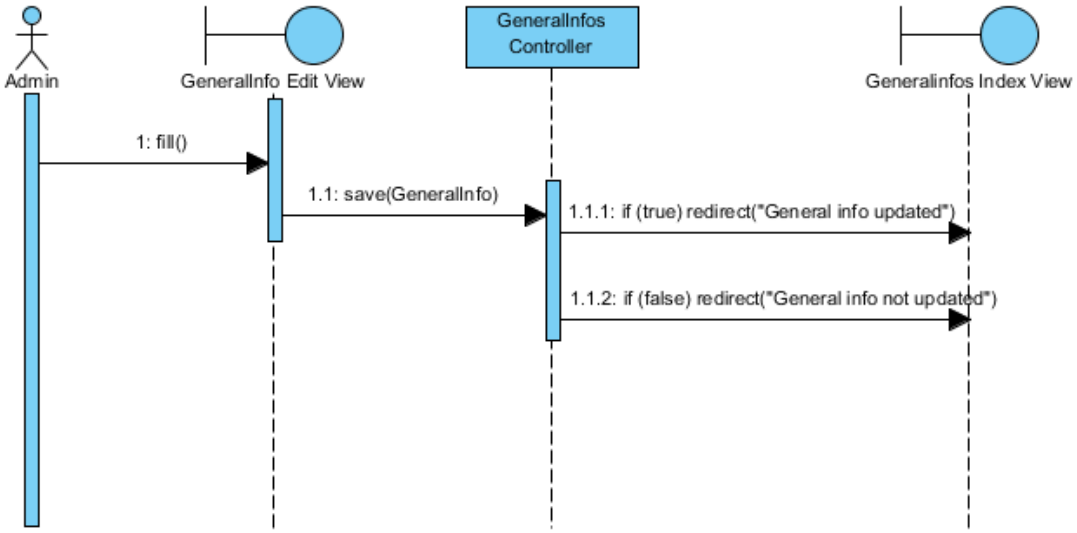


Illustration 30- SEQUENCE DIAGRAM - Edit Study

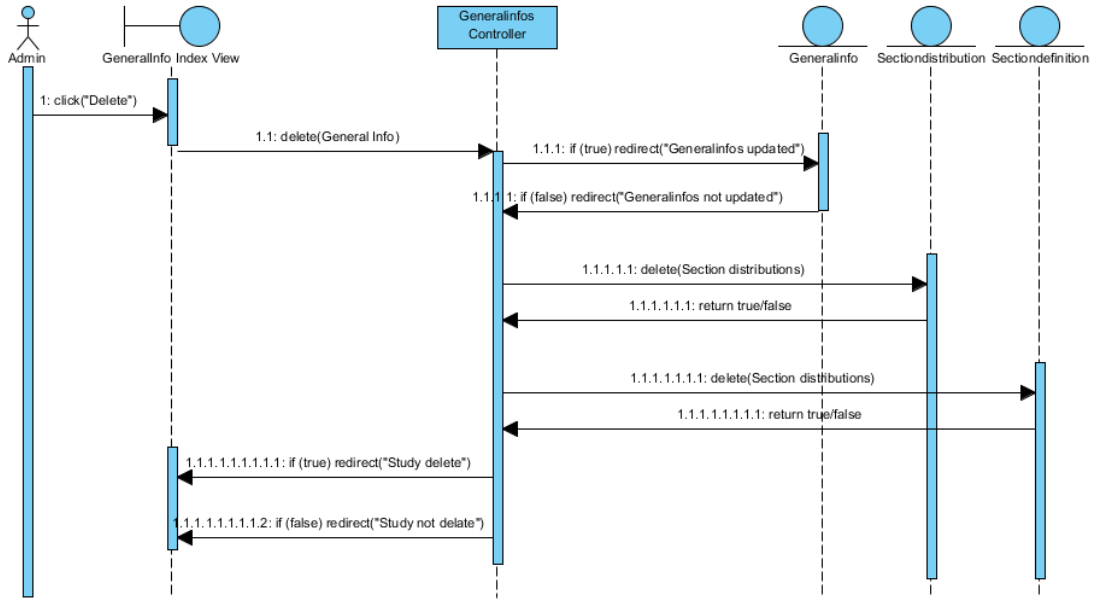


Illustration 31- SEQUENCE DIAGRAM - Delete Study

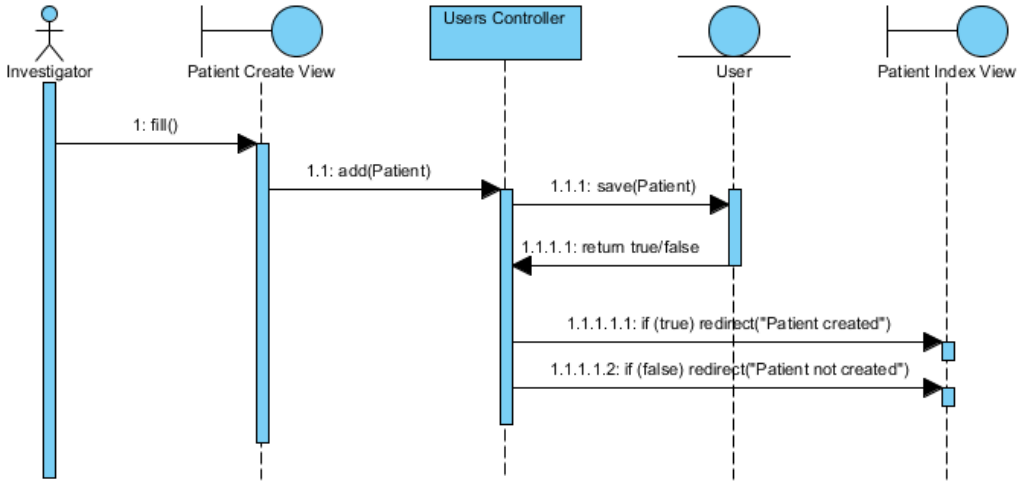


Illustration 32 - SEQUENCE DIAGRAM - Create Patient

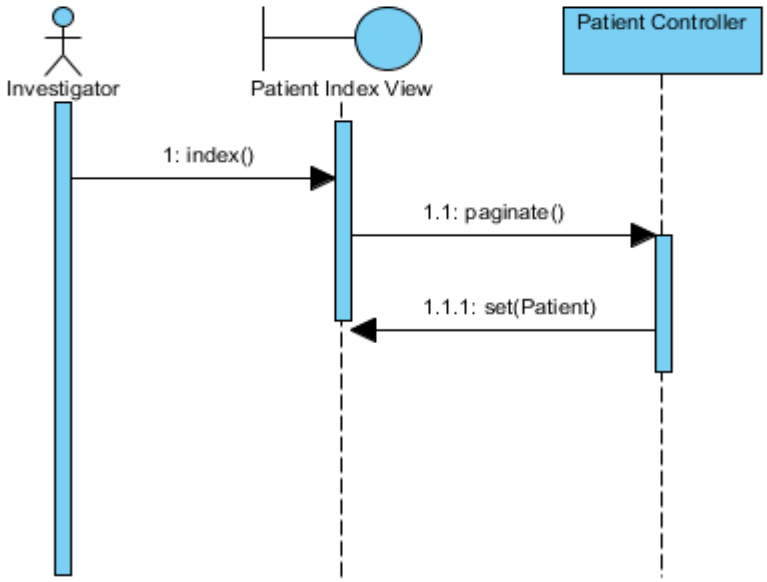


Illustration 33- SEQUENCE DIAGRAM - View Patient

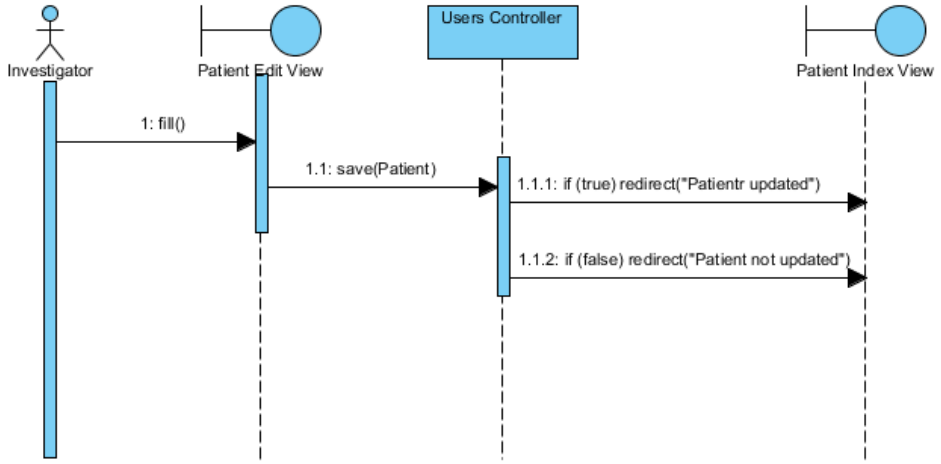


Illustration 34 - SEQUENCE DIAGRAM - Edit Patient

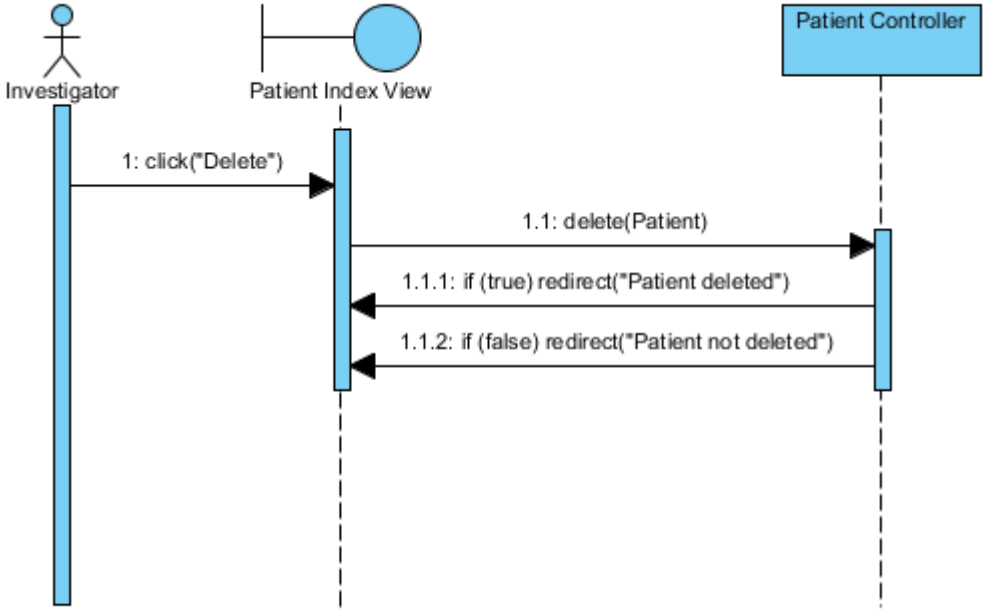


Illustration 35 - SEQUENCE DIAGRAM -Delete Patient